(12) UK Patent Application (19) GB (11) 2 317 476 (13) A

(43) Date of Printing by UK Office 25.03.1998

(21) Application No 9800437.7

(22) Date of Filing 08.07.1996

(30) Priority Data
(31) 50411795 (32) 19.07.1995 (33) US

(86) International Application Data
PCT/US96/11416 En 08.07.1996

(87) International Publication Data
WO97/04412 En 06.02.1997

(51) INT CL[6]
G06F 12/14 1/00

(52) UK CL (Edition P)
G4A AAP

(56) Documents Cited by ISA
US 5379342 A US 5212729 A US 5083309 A
US 4959861 A US 4944008 A

(58) Field of Search by ISA
US Classification: 380/ 25, 4

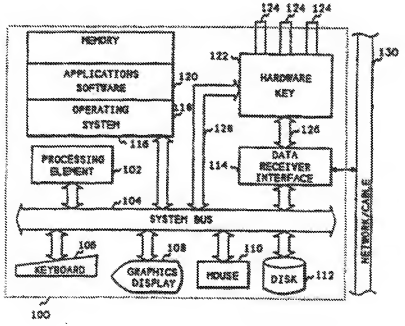
(71) Applicant(s)
Cable Television Laboratories Inc.
400 Centennial Parkway, Louisville,
COLORADO 80027-1208, United States of America

(72) Inventor(s)
Thomas M Williams
Claude T Baggett

(74) Agent and/or Address for Service
Saunders & Dolleymore
9 Rickmansworth Road, WATFORD, Herts, WD1 7HE,
United Kingdom

(54) **Method for protecting publicly distributed software**

(57) A system for protecting software from copying wherein the software to be protected is placed on the computer system in two parts. A first part (120) is stored in non-volatile storage, such as a hard disk or floppy disk within the computer system (100), and a second part is stored and executed in a "hardware key (122)", which is attached to the computer system (100). The second part is stored in volatile RAM (206) and will be erased when electrical power is removed from the hardware key (122), or when the software stops execution. This requires that the second part of the software be reloaded each time the hardware key (122) is powered up. Typically, the second part of the software will be loaded from a network (130), or from a cable network, thus reloading of the second part into the hardware key (122) is a trivial matter, so long as the user is an active subscriber to the network (130) or cable network.



GB 2 317 476 A

(12) UK Patent Application (19) GB (11) 2 361 153 (13) A

(43) Date of A Publication 10.10.2001

(21) Application No 0008276.8

(22) Date of Filing 04.04.2000

(51) INT CL[7]
G06F 17/30 // H04L 9/00 29/06

(52) UK CL (Edition S)
H4P PPEB

(56) Documents Cited
EP 1033854 A2 WO 00/46952 A1 WO 00/01108 A2
US 5915087 A US 5835087 A US 5781550 A
US 5245656 A

(58) Field of Search
UK CL (Edition S) H4P PPA PPEB PPEC
INT CL[7] G06F 17/30 , H04L 9/00 12/28 29/06
Online Databases: WPI, EPODOC, JAPIO

(71) Applicant(s)
Global Knowledge Network Limited
(Incorporated in the United Kingdom)
Suite 94, 2 Lansdowne Row, Mayfair, LONDON,
W1J 6HL, United Kingdom

(72) Inventor(s)
Simon Alan Spacey

(74) Agent and/or Address for Service
Brookes Batchellor
102-108 Clerkenwell Road, LONDON, EC1M 5SA,
United Kingdom

(54) Abstract Title
User security, privacy and anonymity on the Internet

(57) A client accesses a destination sewer over the Internet through an intermediary or proxy sewer. The intermediary sewer receives the client request over a secure encrypted connection, transforms it into a standard request and forwards it to the destination sewer. The request then appears to originate from the intermediary server. Thus logging of client identity and client transactions is prevented. The intermediary sewer transforms the response and further links or references therein into a response from the intermediary site before sending it to the client. Secure email may also be sent without disclosing the sender, receiver or content.

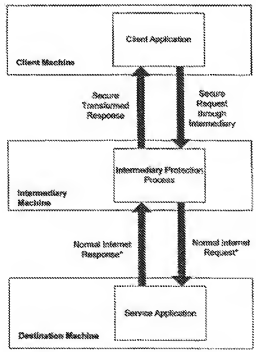


FIGURE 3

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

The claims were filed later than the filing date but within the period prescribed by Rule 25(1) of the Patents Rules 1995.

GB 2 361 153 A

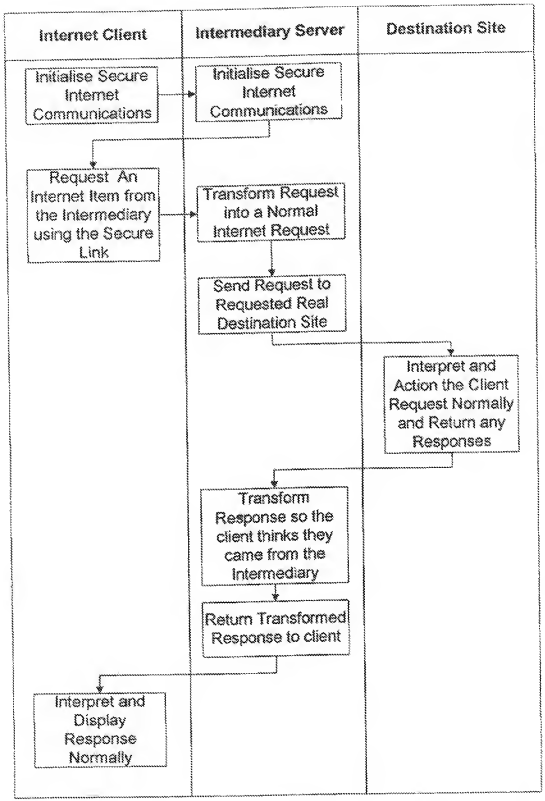

Internet Client
Intermediary Server
Destination Site
Initialise Secure Internet Communications
Initialise Secure Internet Communications
Request An Internet Item from the Intermediary using the Secure Link
Transform Request into a Normal Internet Request
Send Request to Requested Real Destination Site
Interpret and Action the Client Request Normally and Return any Responses
Transform Response so the client thinks they came from the Intermediary
Return Transformed Response to client
Interpret and Display Response Normally

FIGURE 1

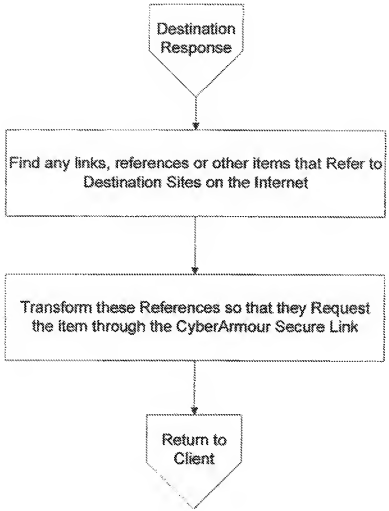
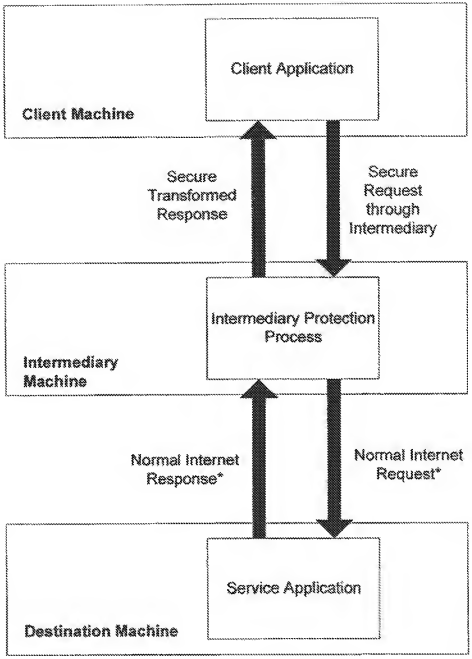

Destination Response
Find any links, references or other items that Refer to Destination Sites on the Internet
Transform these References so that they Request the item through the CyberArmour Secure Link
Return to Client

FIGURE 2



* Potentially Secure Internet Communication Depending on Destination Site and Client Request

FIGURE 3

METHODS AND APPARATUS USABLE WITH OR APPLICABLE TO THE USE OF THE INTERNET

This invention relates to methods and apparatus affording user security, privacy and anonymity on the Internet and World Wide Web.

Hypertext Transfer Protocol (HTTP) is the Internet Application Protocol most widely used on the World Wide Web. HTTP is used by a web browser as a client program to make requests of Web servers through the Internet. A web browser user can request or open a web page by typing in a Uniform Resource Locator (URL) or by clicking on a hypertext link. The browser then sends the HTTP request to the Internet Protocol (IP) address indicated by the URL or link and the requested page is returned. There are many other Internet Application Protocols such as those used for e-mail (SMTP, POP) and file transfer (FTP) as well as proprietary application protocols which are used by Internet applications beyond simple web browsers. HTTP and most other Internet Application Protocols are not secure or encrypted in any way. This means that normal Internet transactions can be easily monitored or tampered with as they pass through the Internet.

When users access the Internet using HTTP or any other Internet protocol, they access the Internet through an Internet provider of some sort. This provider may be their employer, an Internet Café, their own Internet Service Provider (ISP) or some other provider. The user's Internet provider passes the user's request on to the

destination Internet server identified by the URL and associated IP address through routers and other machines that form part of the Internet infrastructure.

User's Internet providers often log the Web Servers and URLs a user visits. These logs are in addition to history files and cookies kept locally on the user's workstation or PC and many users may object to this logging as a breach of their privacy.

In addition to this, the Internet provider and the other routers and machines that form part of the Internet, can often view the entire contents of any of the user's normal insecure Internet transactions. This can include any e-mails picked-up or sent by the user (either using the Web or a mail application) and any forms that the user fills in with personal or financial information on the Internet. The process of viewing Internet transactions as they pass through an Internet provider, router or other machine is called 'sniffing' and is widely available. The ability for Internet providers and other machines to monitor the user's Internet transactions like this further adds to fears of Cyber-Crime and breaches in security and privacy on the Internet.

Anonymity is an additional factor of concern on the Internet. Internet requests often hold in them some information about the requestor. This is often below the Application Protocol Layer and in the case of HTTP Web Browser transactions is at the socket or transport layer. Examples of this information include the Internet Address of the requestor/ user so that the Web Server can return information to them, information about the user's operating system or browser type as well as more

sensitive information. It is possible for destination Internet servers that the user contacts to log this information and use it to breach the user's anonymity.

It is a general object of the present invention to provide methods and apparatus capable of affording security, privacy and anonymity on the Internet. It is also an object of the present invention to provide such methods and apparatus that are compatible with most Internet applications including existing Web browsers.

According to an aspect of the invention there is provided a method of using the Internet which actively prevents any logging by Internet servers, providers, routers and other machines associated therewith of details of destination sites visited by a user or client and preferably, at least, hinders Internet Transaction 'sniffing' on insecure Internet transactions. The method also protects the anonymity of Internet users.

The method may involve a user/ client establishing, preferably through an Internet provider, a connection with an intervening or intermediary site, the intermediary site then provides access to destination sites for the client without the destination sites being logged as having been accessed directly by the client. The only Internet activity of the client that can be logged by any Internet servers, providers, routers and other machines associated therewith is the access to the intermediary site by the client. By using an intermediary site, the method additionally prevents logging by the end destination sites of information as to the identity of the client.

Further, the connection between the client and the intermediary site is preferably a secure, encrypted connection to hinder Transaction 'Sniffing' and further facilitate client Internet privacy. The client to intermediary site connection is preferably secure even if the corresponding client to end destination site would otherwise not be capable of a secure connection. Such a secure connection ensures encryption protection of user requests and responses, information sent through the Internet by the user (this includes the URL of the real destination site the user accesses) and information sent back to users. An example of an encrypted connection is a Secure Socket Layer (SSL) connection. SSL connections provide a public-key encryption framework widely considered to be suitable for commercial exchange and data transferral and are considered secure. SSL encryption capabilities are built in to many Web browser clients today. Using SSL, web browser requests are sent to the intermediary server using HTTPS (Secure Hyper-Text Transfer Protocol) instead of standard HTTP and these requests are transformed and passed on to the destination server using either standard HTTP or HTTPS depending on the secure capabilities of the final destination Web Server.

Preferably in the method of the invention:

1) A client establishes a secure connection with an intermediary site;
2) The client uses the secure connection to send a request for a destination site through the intermediary site;
3) The intermediary site transforms the request into a standard Internet request containing only selected information as to the direct identity of the client;
4) The intermediary site sends the Internet request to the destination site;

5) The destination site returns the requested response to the intermediary site;
6) The intermediary site transforms the response, and preferably any further links or references therein, into a response identified as being from the intermediary site; and
7) The intermediary site, using the secure connection, sends the response back to the client.

The user can read and process the returned destination site information normally and then make a request for another destination site item. To do this the user can simply enter another URL constructed in such a way that it is interpreted through the intermediary site. However, in the case of a Web browser, the user may wish to click on a hypertext link within a viewed web page. Thus, in a practical implementation of the method of the invention, as well as transforming the response into a response identified as being from the intermediary site, the intermediary site finds any references (links or other items) that refer to destination sites on the Internet; and transforms these references so that any future request made by the client using these references is made through the intermediary site. Thus the Web browser client can use the Internet securely, privately and anonymously through the, preferably secure, intermediary server by either inputting URLs directly or by clicking transformed links on web pages in a browser in the normal way to select destination sites through the intermediary server. This transformation process means that Web browsers do not need any configuration changes (such as setting their proxy server to the intermediary server), or any additional software in order for their communications to be 'locked' through the, preferably secure, intermediary server.

Client programs use ports/ sockets to connect to server programs. Port numbers range from 0 to 65535 with numbers 0 to 1023 used for standard services, for example number 80 is used as the default for HTTP and number 443 for HTTPS Web Servers. These defaults do not have to be used and preferably in the method of the present invention non-standard port numbers, i.e. above 1023, are used when establishing connection with the intermediary site. This allows clients to use communications, particularly SSL communications, through existing company or cyber-café firewalls without any reconfiguration. Internet firewalls often stop SSL communications within the standard 0 to 1023 range and are effectively bypassed by using these non-standard port numbers allowing a method, in accordance with the invention, to be used with a variety of firewalls. A method to bypass Internet firewalls using Internet port numbers above 1023 is therefore provided.

Another aspect of the invention provides a method for preventing "Denial of Service attacks" on the intermediary and destination Internet Sites. These attacks are often caused where a malicious client application repeatedly and rapidly sends requests to a destination site but does not wait for the responses. By doing this, the destination site is slowed down because it is continually sending a large number of (potentially large) Internet responses to the malicious client and has no time to service other client's requests. By keeping track of whether clients wait to receive the responses to their requests or not the intermediary server can address these "Denial of Service attacks". Preferably the method comprises holding back the passing on of client requests to the destination site by some period of time, the length of which is related

to the number of times the client has not been present to receive responses for the requests it has sent in the past.

Another aspect of the invention provides a method of sending or receiving an e-mail which actively prevents any logging by Internet servers, providers, routers and other machines associated therewith of details of the destination of the e-mail or its contents. The method may involve the client establishing preferably through an Internet provider a secure, encrypted connection with an intermediary site and sending or receiving an e-mail through the intermediary site. The only activity of the client that can be logged by any Internet servers, providers, routers and other associated machines is the access to the intermediary site by the client.

Another aspect of the invention provides a method of securely storing files on the Internet. The method comprises the client establishing preferably through an Internet provider a secure, encrypted connection with a file storage site through the intermediary server, the client sending a file to the site through the secure connection with the intermediary server and the site storing the file. In the preferred implementation of this method, the intermediary site offers the services of the file storage site itself for the user – removing the need for a second machine and second file transfer. The client can then securely save and retrieve the files by connecting to the secure intermediary site at any time.

According to another aspect of the invention there is provided a method of establishing Internet communication between a client and any normal Internet

destination site by initiating a request containing address information and interposing an intervening site between the client and the destination site, the intervening site acting to ensure that the only recordable information concerning the identities of both the client and destination site is held by the intervening site.

Another aspect of the invention provides a method affording privacy and anonymity on the Internet, the method comprising:

1) A client establishing a secure connection with an intermediary site;
2) The intermediary site offering a range of services to the client; and
3) The client selecting a service.

The services may include using existing external (normal) Internet sites and services while any logging of details of destination sites visited or contents of Internet transactions is actively prevented by the secure layer and the intermediate server, sending or receiving e-mail while any concurrent logging of the destination/ source or contents of the e-mail is actively prevented, and/or storing files securely on the intermediary site. The secure connection established between the client and intermediary site provides communication privacy over the intermediary site's services.

Another aspect of the invention provides a method of establishing an Internet or Internet-type communications link between a client or user site and a destination site for the passage of information therebetween. The method is characterised by interposing an intermediary site between the client or user site and the destination

site. The intermediary site acts as a virtual (and preferably secure) destination site for the client or user site and as a virtual client or user site for the destination site. This is to the extent that all logging entries on the destination site only show the intermediary site as the client or user and all logging entries on the client or user site only show the intermediary site as the destination site.

The methods described herein can improve efficiency and speed of Internet transactions. This can be by the use of compression and other methods. Compression is particularly important for increasing the efficiency of the client connection to the Internet as this is usually relatively slow. Thus the introduction of an intermediary server that compresses transactions as they pass to and from the client is another aspect of the invention. This can be achieved by using compressed SSL communications where the client would otherwise use uncompressed Internet connections.

According to another aspect of the invention there is provided apparatus for performing any one or more of the methods of the invention. Preferably the apparatus comprises a server connected or connectable to the Internet, the server having means to allow a client to establish a secure connection with the server. The server may comprise means to perform any of the steps of any of the methods described herein.

The invention may be understood more readily and various other aspects and features of the invention may become apparent from consideration of the following description.

Implementations and embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a flow chart illustrating the implementation of a method of the invention;

Figure 2 is a flow chart illustrating a general transformation procedure used in the implementation of a method of the invention; and

Figure 3 is a block diagram illustrating an embodiment of the apparatus of the invention in use.

Figure 1 shows the steps taken by an Internet client, an intermediary site and a destination site. A secure Internet connection or link is established between the Internet client and the intermediary site by the Internet client and then the intermediary site initialising a secure Internet communication. In the case of a Web Browser client, a HTTPS connection provides this secure link. The Internet client, using the secure link, requests an Internet item from the intermediary site. A common example of an Internet item is a normal insecure web page from a destination site. The intermediary site transforms the request into a normal Internet request suitable for the destination site to understand - such as a HTTP or HTTPS

request in the case where the destination is a normal Web Server. The normal Internet request, since it is sent by the intermediary site, contains information concerning the identity of the intermediary site and no information or only limited information concerning the identity of the real Internet client. The intermediary site sends the normal Internet request to the destination site containing the Internet item. The destination site interprets and actions the request normally and returns any response to the intermediary site as the site that requested the item. The intermediary site transforms the response to be identified as originating from the request sent to the intermediary site and using the secure link returns the transformed response to the client. The client interprets and displays the response normally. The client can use a similar secure link to make subsequent requests that are similarly processed. The only information relating to Internet activity that can be logged or monitored by a local server or ISP is the accessing of the intermediary site by the client. Importantly, since the client communicates with the intermediary site over a secure link, it is not possible for any Internet servers or the client's ISP to monitor the Internet transaction's contents or even to log the final destination URL the client requested (securely) from the intermediary site.

As well as transforming the response to be identified as originating from the request sent to the intermediary site, the intermediary site performs additional response transformations to Internet items returned from the destination site. The additional response transformations are both client specific and implementation specific and indeed may not be required in some instances and for some application protocols. Figure 2 illustrates an example additional transformation procedure. The

intermediary site locates any links, references or other items that refer to real Internet sites and transforms these so that any requests made for these links are requested via the intermediary site. The intermediary site then returns the transformed response to the Internet client. This 'locks' future requests through the (preferably secure) intermediary site. For example, a Web Browser user can click on a hypertext link within a viewed web page to access a separate web page. The web page is accessed through the intermediary site (following the steps of the method described with reference to Figure 1) rather than directly because the link has been transformed. Direct access, through an untransformed link, would result in the link to the Internet via the intermediary site being broken and normal web access resuming which could be logged or monitored by Internet servers or the user's ISP.

A specific potential transformation of part of a Web site's response is shown below for illustration purposes. A response returned by the destination site to the intermediary site, www.cyberarmour.com, defines a link to another web site, www.gkn.net. The corresponding HTML code segment containing the response is:
<A HREF="http://www.gkn.net">

This line of HTML code is located and transformed to:
<A HREF="https://www.cyberarmour.com:2030/Encrypted:www.gkn.net">

All other references, links and other Internet items would be similarly changed before the response is returned to the client. The word "Encrypted:" and the ":2030" port number are implementation dependent and could be omitted or changed. The

non-standard port number of 2030 has been included here to by-pass Internet firewalls and consequently avoids any potential need for client or firewall reconfiguration. This example transformation is constructed to ensure that when the user clicks on the link generated from the code segment, a request is sent through a secure connection (https://) to the intermediary server (www.cyberarmour.com) bypassing any firewalls (:2030) and requests from the intermediary server the normal HTTP (Encrypted:) Web Server item 'www.gkn.net'.

A preferred embodiment/ implementation, shown in Figure 3, requires no change to the client or destination server components. This implementation is suitable for client applications that have existing secure communication capabilities such as most Internet/ Web Browsers. The client application connects securely to the intermediary server and requests a connection to a destination server through this secure link. The intermediary server transforms the request into a normal Internet request and sends it to the destination server on a "stream" basis. Destination responses are transformed where necessary to force any external links and references to be via the intermediary server (using a general process based on the method described with reference to Figure 2). The transformed responses are also returned to the client on a stream basis.

Using a stream basis the client requests and destination responses are passed/ streamed through the intermediary server as they arrive. Advantageously, no extra client or destination server components or changes are required and no client or destination server speed penalties are seen.

Alternative implementations of the method are also envisaged. For instance, it is possible to pass the data through the intermediary server as a "batch" operation as opposed to on a "stream" basis. The intermediary server would wait to transfer certain whole portions of requests and responses instead of as they arrive. To speed up this process, the intermediary site may cache the transformed requests and responses. Also, multi-stage variations could be used where requests and responses are treated as whole or partial files rather than streams with tasks performed on a batched basis rather than a real-time basis which processes the data as it arrives.

It is also possible to include additional components on the client or destination server machines. These components may be for the provision of secure communication capabilities and/or for performing part of the intermediary site procedures on the client or destination server machine. Various optimisations such as compression and securing the intermediary to destination site connection can also be implemented in this manner. It is also possible to alter some client and destination components to remove the need for link and reference transformations. This includes setting the intermediary server as a web browser's Proxy Server. It is also possible to distribute the intermediary server process across several intermediary servers.

Those skilled in the art will appreciate that there are numerous potential implementations within the scope of the invention as described.

CLAIMS

1. A method affording privacy or anonymity on an Internet-type or other Communications medium, the method comprising:
   a) establishing a secure connection between a client and an intermediary site; and
   b) offering or providing one or more services through or on the intermediary site to the client.

2. A method as claimed in claim 1, wherein the services include using the intermediary site to forward communications between the client and destination sites so as to prevent one or more of the following:
   a) any logging of details of the true destination sites the client has visited by machines capable of monitoring client transactions by means of the secure client-intermediary connection;
   b) any logging of the contents of transactions between clients and destination sites by machines capable of monitoring client transactions by means of the secure client-intermediary connection;
   c) destination sites finding-out the true origin or location of clients by means of formatting client requests to giving the destination site the impression that the intermediary site was the origin of the communication.

3. A method as claimed in claim 1 or claim 2, wherein the services include one or more of the following:
   a) accessing of destination Internet sites by the client through the secure connection with the intermediary site and actively preventing any logging by Internet servers, providers, routers or other machines associated therewith that the destination sites have been visited by the client;
   b) sending or receiving e-mails while any logging of either the destination, source or contents of the e-mail is actively prevented;
   c) storing files securely on the intermediary site;

d) transferring messages between multiple clients connected through the intermediary as in a secure telephone, conferencing, Internet "Chat", "Message Board" service or similar.

4. A method as claimed in any one of claims 1 to 3 and further comprising:
   a) accessing of destination Internet or Internet-type service sites by the client through the secure connection with the intermediary site; and
   b) actively preventing any logging by Internet servers, providers or other machines associated therewith that the destination sites have been visited by the client.

5. A method as claimed in any of the previous claims and further comprising:
   a) establishing the secure connection between the client and the intermediary site;
   b) allowing the client to use the secure connection to send a request to the intermediary site for forwarding to a destination site;
   c) transforming the request into a standard request that can be interpreted by the destination site as originating at the intermediary;
   d) sending the transformed client request from the intermediary to the destination site or a proxy for that site;
   e) receiving the requested response from the destination site at the intermediary;
   f) transforming the destination response into a response identified as being from the intermediary site; and
   g) using the secure connection to return the response back to the original client.

6. A method as claimed in claim 5 and further comprising the step of transforming links and references in the response so that any future request made by the client based on the response from the destination site is made by the client through the intermediary site not directly to the destination site.

7. A method as claimed in any one of claims 1 to 6 and further comprising the intermediary site checking that a client connection remains open to the intermediary throughout a

communication transaction so that destination responses can be delivered to the client and that the client is not attempting an anonymous denial of service attack on the destination site.

8. A method as claimed in any one of claims 1 to 5 and further comprising:
   a) sending or receiving e-mail by the client through the secure connection with the intermediary site; and
   b) actively preventing any logging by Internet servers, providers, routers or other machines associated therewith of details of the client, e-mail content, recipients and sender.

9. A method as claimed in any one of claims 1 to 5 and further comprising sending or retrieving a file by the client through the secure connection with the intermediary site and the intermediary site securely storing or retrieving the file.

10. A method as claimed in claim 9, wherein the intermediary site itself stores the file.

11. A method as claimed in any one of claims 1 to 10 and actively hindering Internet transaction sniffing.

12. A method as claimed in any one of claims 1 to 11, wherein the secure connection is an encrypted connection.

13. A method as claimed in claim 12, wherein the encrypted connection is an SSL connection.

14. A method as claimed in any one of claims 1 to 13 used to allow communication with destination sites where the client is restricted from directly accessing the destination site by a restrictive Internet firewall, proxy server, physical limitations or other apparatus.

15. A method as claimed in claim 14 comprising the intermediary listening for client requests on Internet port numbers above 1023.

16. A method as claimed in any one of claims 1 to 15 and adapted to improve the efficiency and speed of communication transactions by either:
    a) adding compression to the client-intermediary connection
    b) utilising a rapid communications channel between the client and the intermediary so as to reduce overall round-trip or delay times between the client and ultimate destination

17. A method substantially as herein described with reference to Figures 1 to 3 of the accompanying drawings.

18. Use of any of the methods of claims 1 to 17.

19. Apparatus configured to perform any one of the methods of claims 1 to 18.

20. Means to perform any of the methods of claims 1 to 18.

Application No: GB 0008276.8
Claims searched: All

Examiner: Gareth Griffiths
Date of search: 2 April 2001

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.S): H4P (PPA, PPEB, PPEC)

Int Cl (Ed.7): G06F 17/30, H04L 9/00, 12/28, 29/06

Other: Online Databases: WPI, EPODOC, JAPIO

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
X,E	EP1033854 A2	(PITNEY BOWES) whole document	1-5,9,11 at least
X,E	WO00/46952 A1	(FUNDXPRESS) p.3 lines 4 - 20	1,12,13 at least
X	WO00/01108 A2	(PRIVADA) p.5 line 25 - p.12 line 10	1-5,8,11-14,16 at least
X	US5915087	(HAMMOND) abstract	1-5,7-9,11,14,16 at least
X	US5835087	(HERZ) col.31 line 23 - col.48 line 26	1-5,8-12,14,16 at least
X	US5781550	(TEMPLIN) col.3 lines 21-40	1-5,8,9,11-14 at least
X	US5245656	(LOEB) abstract	1-5,9,10,12,14 at least

X Document indicating lack of novelty or inventive step
Y Document indicating lack of inventive step if combined with one or more other documents of same category.
& Member of the same patent family

A Document indicating technological background and/or state of the art.
P Document published on or after the declared priority date but before the filing date of this invention.
E Patent document published on or after, but with priority date earlier than, the filing date of this application.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: 0 421 409 A2

(12) EUROPEAN PATENT APPLICATION

(21) Application number: 90119012.4

(22) Date of filing: 04.10.90

(51) Int. Cl.5: G07F 7/10, G06F 1/00, G06F 15/30, H04L 9/32

(30) Priority: 06.10.89 US 418068

(43) Date of publication of application: 10.04.91 Bulletin 91/15

(84) Designated Contracting States: DE FR GB IT

(71) Applicant: International Business Machines Corporation
Old Orchard Road
Armonk, N.Y. 10504(US)

(72) Inventor: Abraham, Dennis George
5795 Gettysburg Drive
Concord, North Carolina, 28025(US)
Inventor: Aden, Steven George
5641 Mallard Drive
Charlotte, North Carolina 28227(US)
Inventor: Arnold, Todd Weston
2008 Bantry Lane
Charlotte, North Carolina 28213(US)
Inventor: Neckyfarow, Steven William
16 Chevron Drive
Charlotte, North Carolina 28211(US)
Inventor: Rohland, William Stanley
4234 Rotunda Road
Charlotte, North Carolina 28226(US)

(74) Representative: Herzog, Friedrich Joachim, Dipl.-Ing.
IBM Deutschland GmbH Schönaicher Strasse 220
W-7030 Böblingen(DE)

(54) Transaction system security method and apparatus.

(57) An improved security system is disclosed which uses especially an IC card to enhance the security functions involving component authentication, user verification, user authorization and access control, protection of message secrecy and integrity, management of cryptographic keys, and auditability. Both the security method and the apparatus for embodying these functions across a total system or network using a common cryptographic architecture are disclosed. Authorization to perform these functions in the various security component device nodes in the network can be distributed to the various nodes at which they will be executed in order to personalize the use of the components.

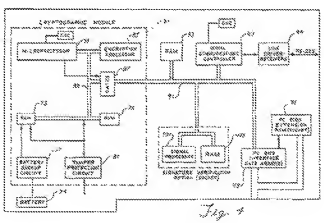


Fig. 1

EP 0 421 409 A2

TRANSACTION SYSTEM SECURITY METHOD AND APPARATUS

This invention relates to security for networks including computer terminals and portable personal data carriers such as IC cards, sometimes called smart cards or chip cards, having an onboard computer and electronic memory for storing data and processing commands.

DESCRIPTION OF THE PRIOR ART

The use of identification cards having computing power and memory built into the card, has been described in the technical literature for some time. Examples are U.S. Patents 4,211,919 to Ugon, and 3,702,464 to Castrucci. A disadvantage of known prior art IC cards that use electrically erasable programmable read only memory (EEPROM) is that the life of an EEPROM is defined by the number of write cycles (e.g., 10,000) before a write failure occurs. Accordingly, the usable life of an IC card using the memory is also limited.

On-card security protection is taught by U.S. Patent 4,816,653. Security is provided in this prior art teaching by having multiple levels of user authorization. Access to a command and to data depends upon who is the current holder of the card, the authority level required to execute a command, and on password data protection contained in the header of each data file.

While providing significantly better user authority checking and security than provided by magnetic stripe identification cards, the above referenced IC cards operate primarily as only semi-intelligent peripheral memory devices. That is to say, the cards respond to read and write command primitives from the workstation, and provide data or record data if the password of the person at the workstation indicates that the person has the authority to perform the requested command. Further, the interface to the prior art IC cards is not well defended. An attack can be made by monitoring the interface while passwords are transferred to or from the card.

Also, the security systems in use with IC cards of the prior art are of a fixed architecture and not easily adapted to differing applications from point of sale to social security or other as of yet unidentified applications. Likewise, when each decision must be referred to the card for processing, a significant number of binary, yes/no responses are provided by the card which may expose the card to attack by unscrupulous persons.

SUMMARY OF THE INVENTION

In accordance with the invention, a highly flexible and secure identification IC card and a distributed authorization system are provided. The invention provides an integrated set of system security capabilities, utilizing the improved identification card of the invention to enhance system component authentication, user identity verification, user authorization and access control, message privacy and integrity protection, cryptographic key management, and transaction logging for audit purposes.

A security system using the invention embodies user authorization in the form of several independent profiles, configurable and programmable by the application owner subsequent to the manufacture of the IC card. Required conditions for the execution of each command are individually programmable by the application owner, using command configuration data. Access to a command is controlled by the content of a user's authorization profile in conjunction with the command configuration data for the requested command.

The user profiles may be downloaded into other security devices in the system for the purpose of controlling use of commands, files, and programs in system component devices, in addition to the IC card itself. The downloaded profile temporarily replaces the authorization profile already active in the other device.

The device command configuration data is not downloaded. The downloaded user authorization profile defines the user's security level and authorizations, while the device command configuration data defines the authorization required by that device to execute a requested command in that device. The same or different commands in other devices to which the user's authorization profile is transferred may have greater or lesser security requirements defined in their command configurations.

The cryptographic keys associated with file and program authorization flag bits in the user authorization profiles that are downloaded into other security system components of an intelligent workstation or other

computer facility, control access to files and programs in that workstation or computer facility.

The command set of the IC card is not fixed. Through use of tables and additional microcode, loaded into the electrically alterable programmable read only memory (EEPROM), new commands can be added to the command set, or existing commands can be replaced with updated versions. Control can also be passed to added microcode in the EEPROM at specific critical points in the IC card supervisor microcode, including initialization, communications, and authorization checking.

The definition of data storage blocks in nonvolatile memory and the read/write access to those data blocks are controlled by security and control information including access prerequisites, stored in the header of each data block in conjunction with the current users authorization profile.

The life of the EEPROM in the IC card is defined by the number of write cycles (e.g., 10,000) before any write failure occurs. For applicable functions, data is written into the memory in such a way as to optimize the total life of the IC card by spreading write cycles across many different storage locations.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a view of the security component devices of the system of the invention;
Fig. 2 is a more detailed block diagram of the IC card of the invention;
Fig. 3 is a block diagram of the circuits of the IC card read write unit;
Fig. 4 is a block diagram of the circuits of the cryptographic adapter card;
Fig. 5 is a block diagram of the software and hardware security components in a workstation;
Fig. 6 is a block diagram of the software and hardware security components of the security processor;
Fig. 7 is a high level flow diagram of authorization checking to execute a command;
Fig. 8 shows content of the user profile and command configuration data tables;
Fig. 9 is a more detailed flow chart of the authorization checking of Fig. 7;
Fig. 10 is a command decode flow diagram;
Fig. 11 shows the structure of data blocks in the memory of the IC card, according to the invention;
Fig. 12 is a summary of the commands for most of the security devices in the network of the invention;
Fig. 13 shows how encryption keys are distributed;
Fig. 14 shows two offline work station logon methods; and
Fig. 15 shows an online work station logon method.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENT

Referring now to Fig. 1, the security component devices are shown in a network environment in which they find utility. The heart of such a network is a host computer 11 which usually will be connected via telecommunication lines to other host computers which are not shown. Host computer 11 performs all the usual data processing tasks for which it is programmed and, in addition, executes the network security processor support program which is the interface between the network security processor 13 and the host computer 11. The network security processor 13 is a small computer which may embody personal computer architecture. Processor 13 may have a display 15, as well as an IC card read write unit 17, according to the invention, and an IC card 19 embodying the invention. Processor 13 operates to provide the inter face for the host computer requests for cryptographic and other security functions and directs the requests to an internal cryptographic adapter card 29.

Communication between host computer 11 and work stations is provided by either direct attach or through a communications concentrator 21. Concentrator 21 is in turn connected to one or more work stations 23 and 25 which may operate together on a local area network. Each workstation will have a keyboard and display and optionally may have a card read write unit 17 for reading and writing information to an IC card 19. In addition, reader 17 may have a signature verification pen 27 for use in capturing the acceleration and pressure dynamics while a holder of card 19 is signing a signature. Processor 13 and work stations 23, 25 may also have a cryptographic adapter card 29 installed into their computer bus. Card 29 has thereon a shielded module 31 which is secure from physical and electrical attempts to read or modify information stored in the memory in module 31.

Each device has the capability to establish a secure session with any of the other devices, or with a remote device which is capable of supporting the secure session establishment protocol. In order for two devices to establish a secure session, they must each contain an identical key, the encrypting key. This requirement guarantees that unauthorized devices cannot establish secure sessions with each other. A result of the secure session process is the establishment of a randomly derived cryptographic session key

known to both devices. Neither the session key nor any other secret data is divulged on the interface between the devices during the session establishment process.

Multiple configurations of system security component devices at the intelligent workstation (IWS) are considered in the system of the invention.

The IWS may utilize only the cryptographic adapter card 29, into which user authorization profiles are downloaded from the host computer and in which high-speed cryptographic functions such as application program encryption are performed. User identification in such an IWS would be accomplished via password entry at the IWS keyboard.

An IWS, utilized primarily in an off-line environment, may have only the IC card read/write unit and the IC card. In this configuration, user identification is effected by entering a PIN on the read/write unit, verification taking place within the user's IC card. The user's authorization profile may be used to control functions performed in the IC card or may be downloaded into the IC card read/write unit to control its functions.

A third configuration, comprising the cryptographic adapter 29, the IC card read/write unit and the IC card, provides all of the functions of the first two configurations. Additionally, it allows the user's authorization profile to be downloaded from the IC card to the cryptographic adapter. A fourth IWS configuration adds to the third configuration the signature verification pen 37, attached to the read/write unit, thereby providing user verification either via PIN or signature dynamics.

Fig. 2 is a more detailed block diagram of the electrical circuits of IC card 19. In Fig. 2, the central processing unit 41 communicates via physical contact with card reader 17 through input/output circuits 43. Connected to the computer bus beside the CPU 41 is random access memory 45, read/only memory 47 and electrically erasable, programmable read/only memory 49.

A number of requests to the IC card require a boolean response, in which the response can have only one of two values. For the purposes of this description, the two values are referred to as TRUE and FALSE. A secure method is used by the programs in the IC card of Fig. 2 to communicate this response.

The method has two very desirable attributes: First, the response is kept secret. Even if the response data is read from the IC card interface, the boolean value of the message cannot be determined. Secondly, if the message is tampered with, as by an adversary who intercepts the message and inserts his own replacement, the act will be detected.

The response is secured through the following cryptographic operation:

1. The requestor generates an eight byte random number, encrypts it under the session key, and sends it to the IC card as part of the request message.
2. The IC card decrypts the random number. If the response value is TRUE, the random number is incremented by one. If the response value is FALSE, the random number is instead incremented by two.
3. The smart card re-encrypts the incremented number under the session key and sends it in the data field of the response message.
4. The requestor decrypts the data, and compares it with the random number he originally sent. If the number is one greater than his original random number, the response is TRUE. If the number is two greater, the response is FALSE. If the number has any other value, the response has been tampered with and is invalid.

Thus, we have accomplished the two goals stated above. The response is secret and cannot be determined by tapping the communications interface, and any attempt to alter the response can be detected.

The random number generator programmed into the IC card uses an 8-byte counter to create different output values each time the algorithm is called. The counter itself is not the random number; it is simply one variable, and is the one used to cause a different value to appear each time.

The counter is in the secure environment of the EEPROM on the IC card, where its value cannot be seen by the user. Thus, it is not important that the counter actually count upward in the conventional sense. What is really important is that it change each time a new random number is generated, and that it step through a very large number of states. Two to the sixty fourth power is the optimal case for a 64 bit counter, but other very large numbers of states are also acceptable under most circumstances.

The EEPROM is nonvolatile, so the counter value is maintained even when the device is powered off. There is one significant problem with EEPROM, however in that each memory cell gradually degrades each time it is written, and will eventually fail, for example, after being rewritten 10,000 times.

If we implement a simple counter, the low order bit changes each time the count is incremented. Thus, we would only be guaranteed 10,000 counts before the device failed. This clearly does not meet the needs of the random number generator.

The improved method of this invention gives more possible values of the counter before the EEPROM

fails. The improved method has a disadvantage in that it does not guarantee all counter values will be different, but it will generate many different values, in a way that cannot be determined from outside the secure environment. It also results in significantly more than the 10,000 cycles possible with the straightforward counter.

The method used updates the counter in a way which will maximize its life. For the EEPROM, this means trying to update each cell of the EEPROM equally often, so all cells will age at an equal rate. This is different from the simple counter, in which low order bits are always updated more frequently than higher order bits.

The method uses the random number itself to index to one of the 64 bits in the counter, then toggles (complements) that bit. The bits of the counter are numbered 0-63, where bit 0 is the low order bit and 63 is the high order bit. The low order 6 bits of the random number are interpreted as a value between 0 and 63, and are used to select the corresponding bit of the counter, which is then toggled. Since the random number generator produces a uniform distribution of values, the 64 bits of the counter are each selected an equal number of times, and none are written more often than any others. Consider the following simplified example, showing a 16-bit counter and the lower 4 bits of the random number.

Counter	Random Number bits	
0000000000000000 (0)	1100	(bit 12)
0001000000000000 (4096)	0101	(bit 5)
0001000000100000 (4128)	1011	(bit 11)
0001100000100000 (6176)	0000	(bit 0)
0001100000100001 (6177)	0111	(bit 7)
0001100010100001 (6305)	....	

Eventually, if the random number values are truly random, the counter would take on all two to the sixty fourth values. It is unlikely that this will happen in reality, but the majority of the values will be attained.

Ideally, the EEPROM would allow toggling of individual bits so that each counter update would result in only one of the 64 bits being written. In most real EEPROMs, however, the smallest unit that can be written is a byte. Thus, when any bit is toggled, the entire byte containing that bit will be written. The result of this is that each of the eight bytes are written 1/8 of the time. The lifetime of the counter is then 8 times 10,000, or 80,000 counts, rather than the 10,000 possible with a straightforward counter.

Fig. 3 shows a block diagram of the circuitry embodied in card reader 17. The computational heart of card reader 17 is microprocessor 51, connected to a bus 53 for communication with other elements of the card reader. Memory for microprocessor 51 is provided in the form of electrically programmable readonly memory 55 and static random access memory 57. Blocks 51, 55, 57, 59 and 65 are enclosed in a secure shielded module with intrusion detection circuitry 59 in order to protect the content thereof. Intrusion detection circuitry is shown, by way of example, in European patent application 90 114 545.8 of common assignee with this application.

In addition to memory, microprocessor 51 is served by real time clock 58. Processor 51 interacts with other devices and the operator, using the following blocks. Communication with the secure cryptographic adapter card 29 in a workstation 25 (or a network security processor 13) and with the standard RS-232 port of a workstation 25 is through asynchronous RS-232 interface 61. The primary communication between card reader 17 and an operator is through operator interface 63, 64 which includes a keypad, an audible beeper, and light emitting diodes. In addition to those operator interface features, the card reader 17 supports a signature pen interface for receiving signals representing the signature of a holder of IC card 19 who wishes to obtain services authorized to the genuine holder of card 19. Pen interface circuitry 65 provides the input ports for receiving change of pressure and acceleration signals representing the signature of the person holding the card. This circuitry and supporting programs are defined in more detail in U.S. Patents 3,983,535; 4,126,829; 4,553,258; 4,724,542; 4,736,445; and 4,789,934, of common assignee with this application.

The IC card 19 itself is read by circuits 67 which include physical and electrical contacts for connecting the circuitry of Fig. 2 to the bus 53 so that computer microprocessor 51 can act in conjunction with the

computer 41 in the card under security programs to transfer information between the card reader and the card.

Referring now to Fig. 4 where the block diagram of the circuits of the cryptographic adapter card 29 are shown, there follows a brief description of each block. The heart of cryptographic adapter 29 is the cryptographic module 31 which provides a tamperproof environment for the encryption processor and storage which contains the cryptographic keys. The cryptographic adapter is controlled by microprocessor 71, using secure memories in the form of random access memory 73 and read/only memory 75. The cryptographic keys are stored in random access memory 73 which is kept active by battery backup circuit 77 and battery 79. In order to thwart an attack on the secure module, battery backup circuit 77 operates under control of tamper protection and detection circuit 81 which detects any attempt to access module 31 by physically attack. The physical and electrical protection of module 31 is set out in greater detail in European patent application 90 114 545.8, of common assignee with this application. Micro processor 71 uses random access memory 83 which is located outside of the secure module 31, in addition to its secure memory. To prevent access to the contents of secure memory 73 and 75 while microprocessor 71 or encryption processor 85 is forming a secure process, gate 87 opens the connection of bus 89 to its outside extension 91 so that any information on bus 89 cannot be read from outside of module 31 at contacts connecting bus 91.

Turning now to Fig. 5, a block diagram of the hardware and software features of a workstation 23 or 25 are shown. A customer application program 111 runs in a workstation 23 or 25, utilizing security utilities 113 and interfaces with the operating system program in the workstation, using a security application program interface. The security utilities provide for such functions as initializing an IC card 19 or enrolling the reference signatures of a user into the memory of the card. Cryptographic function requests from a customer application program 111 are passed through workstation security service supervisor and router 115 to the security server program 117. Security server program 117 provides the program modules and information, including cryptographic keys needed to perform a specific function, to the cryptographic adapter hardware 29 through a device driver program 119. Example program modules include key management module 121, message authentication code verification 123, message authentication code generator 125, and encypher/decypher functions 127, 129.

The keys used for generation of message authentication codes, encrypting of other keys, and ordinary encryption and decryption tasks can be stored in many places in the secure network. Keys are stored on PC disk memory in encrypted form, encrypted under the master key of one of the security devices, cryptographic adapter 29, card reader 17, or IC card 19. Keys are also stored in the nonvolatile memories of cryptographic adapter 29, card reader 17, and IC card 19.

In those configurations where a workstation has both a cryptographic adapter 29 and an IC card reader 17, security functions relating to the IC card or card reader are requested by customer application program 111, pass down through the various program interfaces through cryptographic adapter 29 to card reader 17. In those configurations where a workstation has only a card reader and no cryptographic adapter, the card reader is connected to the personal computer of the workstation by asynchronous communication interface 81, shown in Fig. 3, which is represented as a communication line in Fig. 5.

Referring now to Fig. 6, a more detailed block diagram of the circuits and programmed functions, residing in network security processor 13, are shown. Network security processor 13 is based upon a personal computer architecture running a special security operating system which prevents the computer from performing ordinary personal computer functions, thereby enhancing security. The security operating system is based upon an IBM Personal Computer Disk Operating System 141 and modified by a multi-tasking program 143. One of the tasks running under multi-tasking program 143 is a host server module 145. Server 145 manages the communications between the network security processor 13 and the host computer 11 through a channel task program 147 and a host channel adapter 149. Of particular importance is another task in the form of security server program 151, performing functions complementary to the security functions performed by the security server 117 in the workstation shown in Fig. 5. This is accomplished by the cryptographic adapter task program 153 and cryptographic adapter device driver program 155 which provide the interface to the cryptographic adapter 29, installed in the personal computer bus of network security processor 13. The IC card reader 17 and its associated IC card 19, attached to the network security processor 13, are used to control access to the network security processor for initializing the security processor, operator services, and maintenance etc. Another function served by the card reader is to accept parts of master keys in secure fashion in order to initialize the security processor. That, after the master key entered in parts, is used to generate other keys for distribution to other devices at other nodes in the secure network.

The directory server task 157 contains the pointers and program routines to allow the security server to

access encryption keys and other information needed to perform its cryptographic functions, interfacing with PC DOS file access method programs 159. Log server 161 also is a task which provides for the auditing functions needed by the system. At the top of Fig. 6 are shown blocks 163 which provides installation services programming, 165 which provides initial program loading services, and 167 which provides operator interface programmed functions.

Fig. 7 is a high level view of the processing method which decides whether a user is authorized to execute a particular command. Each test references one or more tables, which are shown attached to the corresponding processing step.

The first step 171 checks whether the command is a universally authorized command. Universally authorized commands listed in table 173 are a fixed, predefined set of commands that are necessary for all users in all situations. They are always allowed, regardless of the user's authority. None of these commands are security-related.

The next two steps 175 and 177 are actually performed together, but are shown separately for clarity. These involve checking whether the current user is authorized to execute the particular requested command. A user's authority is defined by the contents of a related user profile in the table of user profiles 179. The requirements for execution of the selected command are defined in command configuration data table 181 by the execution prerequisites for that command. These two items of information from the tables are examined to determine if the user is permitted to execute the command. These steps are set out in more detail in Fig. 9.

If the user has the authority to execute the selected command, there is one additional step 183 that still must be performed. A programmable table 185 contains a list of dates defined as holidays, and most commands cannot be executed on a holiday. This provides an additional level of security. If the current date is listed as a holiday, all commands except the universally authorized commands are disallowed.

Once it has been determined that the user is authorized to execute a command, the command is decoded at block 187, using the command decode tables 189 shown in more detail in Fig. 10. The command is executed at block 191 of the flow diagram, after which control of the IC card or other security device returns to wait for the next command.

Fig. 8 represents the relationship between user authorization profiles 179 and command configuration data 181, as they are utilized within the IC card to securely limit the use of commands, as programmed by a designated authority of the application owner.

Each of the user authorization profiles 179 contains a command authorization flag bit 197 for each command used in any of the system security devices. If the flag bit is not set, then the user is not authorized to execute the corresponding command.

User authorization profiles 179 also contain some number of file and program authorization flags 199. When the user profile is downloaded into a workstation cryptographic adapter, each file authorization flag bit is associated with a cryptographic data key used for encrypting or decrypting a specific file. Similarly, the program authorization flag is used to control access to specific programs.

Other data 199, in the user authorization profiles 179, specify a level of authority in the exercise of commands, time of day and day of the week limits, expiration date for the user authorization, and other user flags indicating the mode for the identification of the user.

The command configuration data 181 is independent of the user authorization profile, but consists of a number of prerequisite conditions and authorizations for each command. There is a unique set of command configuration data for each of the system security devices in the system.

Fig. 9 is a detailed flowchart showing exactly how the authorization checking of Fig. 7 is performed. The first step 201, as in Fig. 7, is to check the table 173 of universally authorized commands. If the command is in this table 173, remaining steps are bypassed and the command is automatically authorized.

At block 203, the user's user profile 179 is retrieved and, at block 205, the command configuration data 181 for the selected command is retrieved. These are used in performing most of the remaining checks. If the command unavailable flag is found, at block 207, to be set in the command configuration data 181, the command is not authorized and the remaining steps are bypassed.

If the secure session required flag is found at step 209 to be set in the command configuration data 181, the command is not authorized unless a secure session is determined at block 211 to be in effect with the sender of the command. This has the effect of allowing the command only if the sender of the command has been verified as an authentic system component or device as for example, an IC card or cryptographic adapter etc. A secure session cannot be established between two components that do not share certain common cryptographic keys installed by the owner.

If block 213 determines that the initial verification required flag is set in the command configuration data 181, the user must have verified his identity at some time during the current session, or the command will

not be allowed. This is tested at block 215. He may have verified his identity by entering his PIN, or by using signature verification, or some other external means. The methods he can use for verification are controlled by the verification method identifier in his user profile.

If the pre-execution verification required flag is set (block 217) in the command configuration data 181, the user must re-verify his identity before each time the command is executed. Block 219 tests whether the user has re-verified his identity in order to use this command. If this flag is set and the user has not re-verified for execution of the command, it will not be allowed.

Block 221 determines if the disable time limits flag is set in the command configuration data 181. If it is set, the time of day, date, and day of week checking at block 223 is bypassed for the command. If the flag is not set, the time of day limits, expiration date, and valid days of week fields in the user profile are compared to the current time, date, and day of week to determine if the command is allowed. If any of these are not satisfied, the command is not allowed and further checks are bypassed.

If the current date is found at block 225 to be listed as a holiday in the programmable holiday table, the command is not allowed. The user's authority level in his user profile is compared at blocks 227, 229 and 231 to the authority level required to authorize the selected command, which is contained in the required authority level field of the command configuration data 181. If the authority exact match flag is set in the command configuration data, the user's authority level must be exactly equal to the required authority level for the command to be allowed. If the authority exact match flag is not set, the user's authority level must be greater than or equal to the required authority level for the command to be allowed.

Each user's user profile contains a set of command authorization flags defining which commands that user is excluded from executing. If the requested command is not authorized in the user's command authorization flags, execution is not allowed by the logic in block 233.

Each user's user profile contains a verification failure count which counts the number of consecutive verification failures, either by PIN or by signature verification, or another external means. Each profile also contains a programmable verification failure limit, defining the number of consecutive verification failures the user is permitted before he is locked out. At block 235, the user's verification failure count is checked to see if it is greater than or equal to his verification failure limit, and if so, the command is not allowed.

Referring to Fig. 10, the method of command decoding in the IC card is shown. This method employs two command decoding tables: one 241 in the microprocessor ROM, which is fixed, and another 243 in the electrically erasable programmable read only memory (EEPROM), which is programmable. The table 241 in ROM defines the default subroutine address to be called for each of the possible commands. The table 243 in EEPROM can be loaded with new addresses, which will override those in the ROM table. The method allows one to load new commands into EEPROM, or to load replacements for commands in the ROM, and to use the EEPROM table to cause these downloaded commands to be executed in place of the commands in the ROM. Whenever a command is to be executed, the address is first read from table 243 in EEPROM. If block 245 in Fig. 10 determines that the address from the table 243 is not zero, it is used as the address of the subroutine to process the requested command. If the address is zero, an address is read from the table 241 in ROM and the address read from ROM is used for the subroutine to process the command. Thus, any ROM command subroutine can be replaced by inserting a non-zero address into the table 243 in EEPROM.

Fig. 11 shows the format used on the IC card to store data blocks. Data blocks are a general purpose means for defining and managing user or system data areas in the IC card non-volatile memory. Data is written to the blocks and read from the blocks. There are many options and features to keep the data secure from attacks.

251 in Fig. 10 shows the overall layout of data blocks in the IC card EEPROM memory. The low portion of the memory is reserved for information that is not related to the data blocks. All memory above this fixed, predefined data is available for the definition of data blocks. They are allocated in contiguous segments of the memory. The first data block defined occupies memory starting immediately after the fixed data, the second block defined immediately follows the first, and so on.

253 shows the structure of a single data block. Each block consists of two parts, a header and a data area. The header contains control information related to the block, and the data area contains the data which is written to and read from the block. The information in the header is defined when the block is allocated. The data area is of a fixed size once the block has been defined by the one of the users.

255 shows the contents of the block header. The block ID is an eight byte field used to identify the block. It is passed to the card with all data block commands in order to identify the block of interest. Any eight byte value is permitted. The token is a secret value used to authorize access to the data in the block. The user must pass the correct token to the IC card with each data block command in order to be granted access to the block. The token is similar to a password for access to the block. It is defined by the user at

the time the block is allocated. In order to protect the block ID and token from disclosure, they can be encrypted under the session key when they are transmitted to the IC card.

The data length field in 255 defines the number of 8-byte paragraphs in the data area of the block. A value of 1 indicates that there are 8 bytes in the data area, a value of 2 indicates there are 16 bytes, and so on.

A checksum is stored in the header 255 to allow verification of data integrity in the data portion of the block. The checksum is calculated from the data each time it is written, and the checksum is verified each time the data is read. If the checksum indicates there is an error in the data read, the data is still returned to the requester, but a warning code is returned to inform the requester of the error condition.

The header 255 contains read authorization flags and write authorization flags for each user profile. Each of the possible IC card users can be given read only access, write only access, read/write access, or no access to each data block individually.

The header also contains a minimum authority level which is compared with the authority level in the user's profile. The user's authority level must be greater than or equal to the minimum authority level in the block header in order for that user to be granted access to the block.

A set of flags 257 in the block header 255 defines various security features for the block. The verification required flag, if set, indicates that the user must have verified his identity before he can be granted access to the block. The user can verify his identity with PIN or with signature verification or another external verification means. A hidden block flag, if set, indicates that the block will not be listed when the user requests a list of the blocks that exist on the IC card.

A secure session required flag, if set in 257, indicates that the block cannot be accessed from a device unless that device has a secure session in effect with the IC card. A session key encryption required flag, if set in 257, indicates that all data transmitted to the card for writing in the block, or transmitted from the card when read from the block, will be encrypted using the session key established between the IC card and the device with which the secure session has been established.

If the secured block flag in 257 is set, the block token must be passed to the IC card encrypted under a cryptographic key. The IC card will decrypt the token using the specified key, and compare the decrypted result with the token stored in the block header 255. Access to the block will be denied unless the decrypted token is correct. This ensures that the block can only be accessed if the requester knows the correct token, and possesses the correct cryptographic key. This has the effect of protecting the data from either read or write access unless the requester knows the correct secret key.

A typical method for protecting data using encryption is to encrypt the data itself when it is stored. The correct key must be used when it is read back and decrypted in order to retrieve meaningful data. This protects against reading by those who do not possess the secret key. It has two disadvantages, however. First, it requires the overhead of encrypting and decrypting the data, which can be time consuming for large data blocks. Secondly, it only protects the read operation. The data can still be overwritten by someone who does not possess the key, although the data written might not be meaningful.

The secured block concept employed in the IC card described here is a superior alternative to simple encryption of the data in the block. It requires far less encryption overhead, and also protects both reading and writing of the data block. The method encrypts the block token when it is sent to the IC card, rather than encrypting the data itself. The IC card decrypts the token, and if the user does not possess the correct cryptographic key, the IC card will recover a token value that does not match the token stored in the block header. Access to the block, either in read or write mode, will then be denied. Only encryption of the eight byte token is required.

Note that storing the data in encrypted form is not required in the IC card. The data is stored in the EEPROM, which is a secure environment. The only need for encryption of the data is when it must be protected as it passes over the interface to the IC card. For that purpose, the IC card can accept data encrypted under the session key for the write operation, and can encrypt outgoing data under the session key for the read operation.

A summary of the classes of default commands that can be executed by the security component devices is shown Fig. 12. In the IC card, for example, additional and different commands can be downloaded to the IC card device in order to perform additional functions as they are found to be needed.

Fig. 13 describes a method of cryptographic key initialization of the system security component devices of the system of the invention. Reference is made also to Fig. 1.

The host CPU 11 responsible for overall key management for the system or network contains in its network security processor 13 a host master key under which the master node keys for all other network node devices are encrypted. The host master key is generated manually by a privileged and responsible individual (security administrator) in a highly secure and protected environment.

The host master key may be entered into the network security processor 13 in several different ways. Using IC cards 19 with the highest level of authority in its user authorization profile, the security administrator generates master key parts on his IWS 25, incorporating a cryptographic adapter 29 and IC card read/write unit 17, and then enters the host master key parts into the network security processor 13, using its attached IC card read/write unit 17. This is accomplished through the use of commands defined under a common cryptographic architecture specifying the cryptographic structure, commands and operation of all system security component devices in the system of the invention. The common cryptographic architecture is described in great detail in co-pending patent application US Serial Numbers 231,114 (European patent application 88 308 068.9), 233,515 (European patent application 88 308 079.6), 237,938 (European patent application 88 308 071.3), 238,010 (European patent application 88 308 070.5) and 344,165 (European patent application 89 105 905.5). Alternatively, the security administrator may enter the host master key into the network security processor 13 directly through the PIN pad keyboard of the IC card read/write unit under the authorization profile loaded into it from the security administrator's IC card.

The next step in the process of cryptographic key initialization of the system or network is to generate network node master keys encrypted under the host master key. Toward maximization of security in the transportation of the node master keys from the network security processor 13 to the remote network nodes, the node master keys are generated in parts and each part written into the secure memory of separate IC cards 19. This step is shown at block 311 in Fig. 13. Only after the key parts are sequentially imported from the separate IC cards 19 containing the key parts to another system security component device, and cryptographically assembled, is the node master key usable. Importing or loading the key parts into other node devices is shown at steps 313 and 315 in Fig. 13.

After all system or network nodes have been so initialized with master node keys, node key encrypting keys may be generated by the central network security processor 13 under control of the key management application program running on the host CPU 11, and encrypted under the specific node master key which is held within a secure key directory in the network security processor 13. Other secondary keys such as data keys for specific purposes, may then be transported through the system facilities, encrypted under a nodes key encrypting key. This step appears at block 317 in Fig. 13. Transportation of these keys is effected through the host computer 11, as shown by step 319, to the system or network communications facilities. The secondary keys are downloaded at steps 321 and 323 in a secure session to each node represented by a security component device such as a cryptographic adapter 29 or an IC card read write/unit 17. Depending on the key management structure in the application, the need for data keys to be held in common between the central network security processor 13 and other system or network nodes, and the level of key management control delegated to the system or network node, the secondary keys may include data keys for safeguarding the files and programs of the node device. Alternatively, these keys could be generated locally at the node under the node master key.

Fig. 14 illustrates the off-line intelligent work station (IWS) logon procedure in the system of the invention, using the system security component devices described in Figs. 1 through 5 of this invention. Reference is made to Fig. 1 for devices identified therein.

When the user inserts the user's IC card, step 325, into the IC card read/write unit 17, those two devices establish a secure session between them in step 327. This action occurs transparently to the user, is built on the existence of a cryptographic processor in both devices, and results in a unique session key. When the secure session has been established, the cryptographic adapter 29, if it is present in the IWS, is advised by the IC card read/write unit 17 that the secure session has been established. At step 329, action is then initiated to establish a similar secure session between the IC card 19 and the cryptographic adapter 29. At the conclusion of that action, the IC card is in secure session with both of the other devices. The cryptographic adapter will attempt to establish a secure session with the reader at initialization of the cryptographic adapter. In the process of establishing secure sessions among these devices, the authenticity of each device is verified.

The next step in the logon procedure is to verify the identity of the user person to the IWS. Fig. 14 illustrates two methods of user verification: one based on the use of a secret pin verification number (PIN); and another based on the comparison of dynamic signature pattern data of a verification signature with that of reference signatures prerecorded in the IC card. Because the latter procedure is inherently less susceptible to compromise and more costly to implement than the former, the choice between the two reverts to a value judgment for each application.

The PIN verification is initiated at step 331 by a prompt to the user to enter the user PIN on the PIN pad of the IC card read/write unit 17. Within this unit, the PIN and a random number are encrypted using the session key, and passed across the protected interface to the IC card 19. Within the IC card 19 at step 333 the received quantity is decrypted, the random number is separated from the PIN, the PIN is verified

against the user PIN stored in the IC card. Based on the result of the verification attempt, the random number is incremented by either a 1 or a 2 and encrypted to provide a protected response to the IC card read/write unit.

Alternatively, signature verification is initiated at step 335 by a request from the cryptographic adapter 29 to the IC card 19 to download the user's signature reference data. In the IC card, the signature reference data is read from secure memory, encrypted, and passed through the IC card read/write unit 17 to the cryptographic adapter 29, where it is decrypted at step 337 and held in memory.

The user is then prompted at step 339 to write a verification signature, and using the signature verification pen 27 attached to the IC card read/write unit 17, the user writes a signature. The analog signals from the pen are digitized and encrypted and passed across its protected interface to the cryptographic adapter 29, where the signature data is decrypted and placed in memory. Within the cryptographic adapter at step 341, the dynamic signature verification algorithms, e.g. as described in US Patent 4,724,542 are invoked to effect a confident match of the verification signature data against the multiple reference signature data sets.

Thus far in the off-line logon process, the authenticity of the security component devices have been validated and the user person's identity has been verified to the IWS. It remains to establish, within the security component devices of the IWS, the authorization to access IWS resources within time-of-day/day-of-week limits and more specifically to exercise the command set of the device, to utilize files and programs within the IWS.

Requests from the card reader 17 and the cryptographic adapter 29 to the IC card 19 would result at step 343 in the downloading of the user authorization profile to the IC card read/write unit 17 and to the cryptographic adapter 29. Then, as described in detail with Figs. 7, 8 and 9, the user verification profile, the command configuration data and cryptographic keys combine at steps 345 to control the use of commands, files and programs throughout the session.

On logoff, the user authorization profile that had been downloaded from the user's IC card to the cryptographic adapter is removed at step 347, and the cryptographic adapter reverts to its default profile.

Fig. 15 illustrates the intelligent work station (IWS) on-line logon procedure, utilizing the cryptographic adapter as the only system security device in the IWS.

Through the communications facilities in the IWS 25 and the host CPU 11, a secure session is established between the cryptographic adapter 29 in the IWS and the network security processor 13. As it is in the off-line logon procedure described in connection with Fig. 14, the establishment of the on-line secure session is transparent to the IWS user.

User identification with this IWS configuration is initiated at step 351 by the entry of the user's password at the IWS keyboard in response to a prompt message. The logon password is verified at step 353 in the network security processor 13 against its directory of authorized users. A positive verification results in the retrieval at step 355 of the user authorization profile from the directory. The profile is then encrypted under the session key created for the session and the encrypted profile is downloaded at step 357 to the cryptographic adapter 29 in IWS 25.

The IWS 25 may then continue to operate in an on-line mode with the host CPU 11 as a continuation of the same secure session or under a subsequent secure session. The IWS may also revert to an off-line mode represented in step 357 in which the user authorization profile downloaded from the network security processor 13, the common configuration data resident in the cryptographic adapter 29 of the IWS 25, and (block 358) the secondary cryptographic keys previously established in the cryptographic adapter all serve to control the use of commands, files and programs in the IWS. As in the off-line case, logoff at step 359 results in the removal of the downloaded user authorization profile and substitution of the default profile.

While the invention has been described with reference to a preferred embodiment thereof in the form of a transaction security system including an IC card, it will be apparent to those skilled in the art of computer system design that the principles, methods, and apparatus of the invention can be applied in other environments to enhance the security and prevent fraud.

Claims

1. A security device comprising:
a data processor;
memory connected to said processor;
data input and output means connected to said processor;
secure session establishing means programmed into said security device for controlling said processor to

establish a secure session with another device;
an authorization profile stored in said memory, said profile defining the authority of a user of said security device to cause said processor to execute programmed commands;
transfer means for transferring at least part of said authorization profile from said security device to said another device for controlling said device in accordance with said authority of said user defined in said authorization profile.

2. A security device comprising:
a data processor;
memory connected to said processor;
data input and output means connected to said processor;
secure session establishing means programmed into said security device for controlling said processor to establish a secure session with another device;
means for receiving at least part of an authorization profile stored in a memory of said another device, said profile defining the authority of a user to cause said processor to execute programmed commands.

3. The security device of claim 1 or 2, wherein said security device is an IC card, or a host computer, and said another device is an IC card reader, or a computer work station.

4. The security device of claim 1, 2, or 3, wherein said authorization profile defines the authority of said user to execute a command at a particular time, and/or day and/or between particular times of day.

5. The security device of claim 1, 2, 3, or 4, wherein said authorization profile contains a plurality of command flags, each command flag defining the authority of said user to execute a command.

6. The security device of claim 1, 2, 3, 4, or 5, wherein said authorization profile contains a plurality of access flags, each access flag defining the authority of said user to access a data file.

7. The security device of claim 1, 2, 3, 4, 5, or 6, wherein said authorization profile contains a plurality of program flags, each program flag defining the authority of said user to execute a program.

8. The security device of anyone of claims 1 to 7, wherein said authorization profile contains a user authorization level.

9. The security device of anyone of claims 1 to 8, wherein said authorization profile contains a user ID, a personal identification number, and an identity verification method identifier.

10. A security device, especially an IC card, comprising:
a data processor;
protected programmable memory connected to said processor;
data input and output means connected to said processor;
an authorization profile stored in said memory, said profile defining the authority of a user of said security device, especially said card to cause said processor to execute programmed commands;
means for receiving an authorization profile created by an authorized person and storing said received authorization profile into said memory to be used in place of said stored authorization profile.

11. The IC card of claim 10 further comprising:
data blocks in said memory, each data block having a header, said header containing memory access prerequisites; and
means for comparing said access prerequisites with the authorization profile of said user.

12. The IC card of claim 11 wherein one of said access prerequisites is an authority level and further comprising:
means for comparing said authority level with an authorization level stored in said users authorization profile, and/or
wherein one of said access prerequisites comprise a read flag and a write flag for each user and further comprising:
means for allowing read access to said memory only if said read flag is set and allowing write access to said memory only if said write flag is set, and/or
wherein one of said access prerequisites is a secure session required flag and further comprising:
means for allowing access to said memory only if said IC card is in a secure session with another device which is requesting access to said memory.

13. A security device comprising:
a data processor;
protected programmable memory connected to said processor; data input and output means connected to said processor;
a plurality of commands for controlling said processor stored in said memory, each command having a plurality of programmable execution prerequisites stored in said memory.

14. The security device of claim 13 wherein one of said execution prerequisites is an established secure

session between the devices affected by the command, whereby the command to which it relates will not be executed unless a secure session has previously been established, and/or

wherein one of said execution prerequisites is an initial user verification whereby the command to which it relates will not be executed unless the identity of the user requesting the execution of the command has previously been verified during a current session, and/or

wherein one of said execution prerequisites is a pre-execution user verification whereby the command to which it relates will not be executed unless the identity of the user requesting the execution of the command has been verified specifically for each execution of said command to which it relates, and/or

wherein one of said execution prerequisites is time, whereby a command to which it relates will not be executed unless the time and date are within the limits authorized during which a user requesting execution of said command is authorized to execute said command, and/or wherein one of said execution prerequisites is an authorization level, whereby a command to which it relates will not be executed unless a user requesting execution of said command has an authorization level at or above a specified level.

15. Method of communicating a secure boolean response, especially for use in a device of anyone of the claims 1 to 14, comprising the steps of:

a) generating a random number in a security device;

b) encrypting said random number under a key;

c) sending said encrypted random number to another security device;

d) decrypting said encrypted random number in said another security device;

e) modifying said random number by a first function if said response is true;

f) modifying said random number by a second function if said response is false;

g) encrypting said modified random number;

h) sending said encrypted modified random number to said first security device;

i) decrypting said encrypted modified random number at said first security device; and

j) comparing said modified random number with said random number to determine the response.

16. Method of changing a value used in the generation of a random number, especially for use in a device of anyone of the claims 1 to 14, comprising the steps of:

a) generate a first random number;

b) using a portion of said random number to select a bit of said value;

c) inverting said bit;

d) repeat steps a, b, and c to generate a second random number.

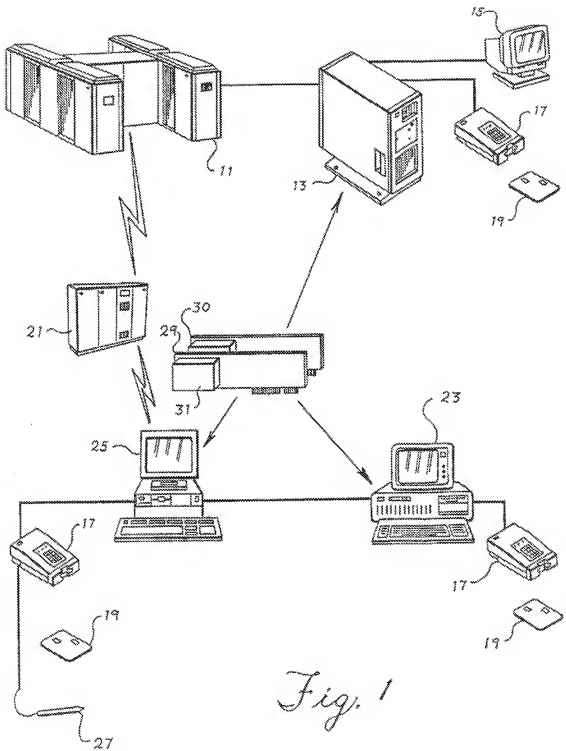


Fig. 1

Fig. 2

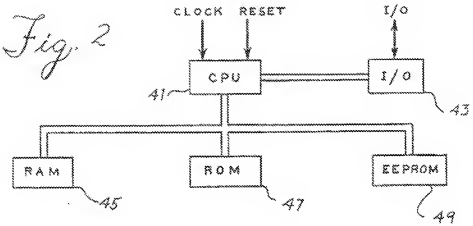
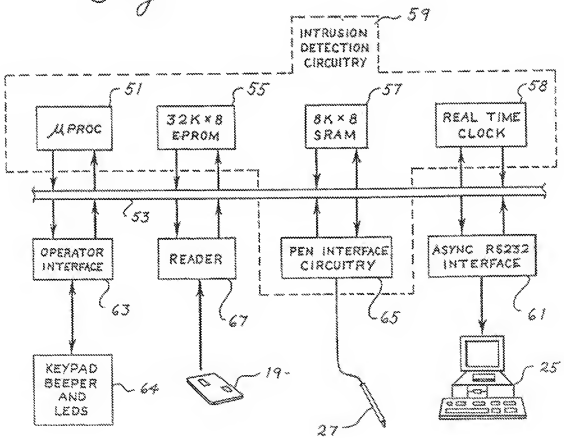

CLOCK
RESET
I/O
CPU
I/O
41
43
RAM
ROM
EEPROM
45
47
49

Fig. 3


59
INTRUSION DETECTION CIRCUITRY
51
μPROC
55
32K×8 EPROM
57
8K×8 SRAM
58
REAL TIME CLOCK
53
OPERATOR INTERFACE
63
READER
67
PEN INTERFACE CIRCUITRY
65
ASYNC RS232 INTERFACE
61
KEYPAD BEEPER AND LEDS
64
19
25
27

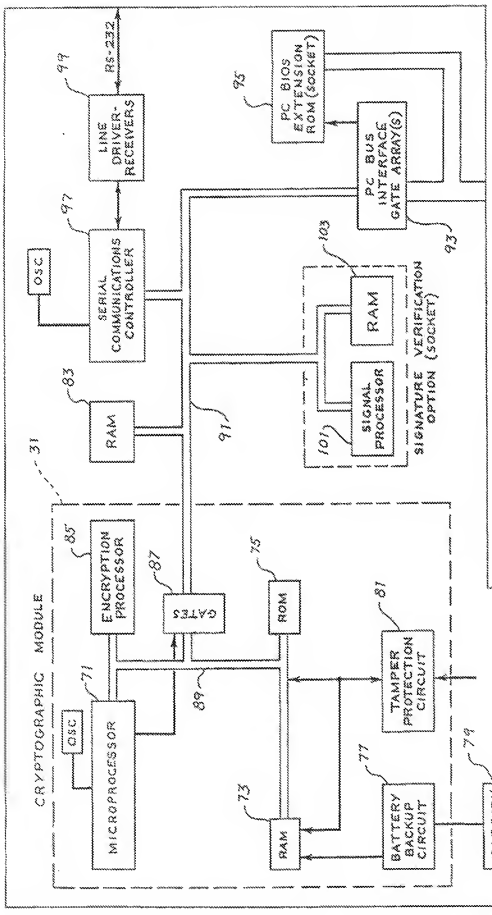

CRYPTOGRAPHIC MODULE
31
MICROPROCESSOR
71
OSC
ENCRYPTION PROCESSOR
85
GATES
87
ROM
75
RAM
73
BATTERY BACKUP CIRCUIT
77
BATTERY
79
TAMPER PROTECTION CIRCUIT
81
89
RAM
83
OSC
SERIAL COMMUNICATIONS CONTROLLER
97
LINE DRIVER-RECEIVERS
99
RS-232
91
SIGNATURE OPTION
SIGNAL PROCESSOR
101
RAM
103
VERIFICATION (SOCKET)
PC BUS INTERFACE GATE ARRAY(S)
93
PC BIOS EXTENSION ROM (SOCKET)
95

Fig. 4

Fig. 5

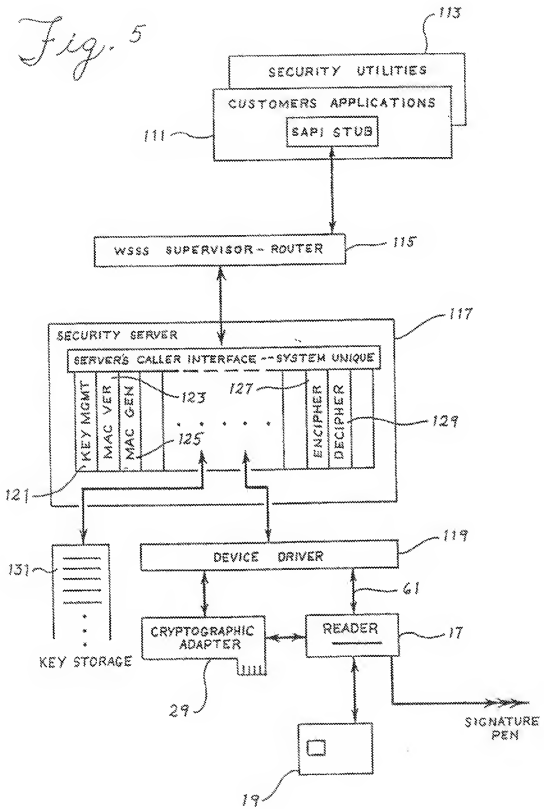
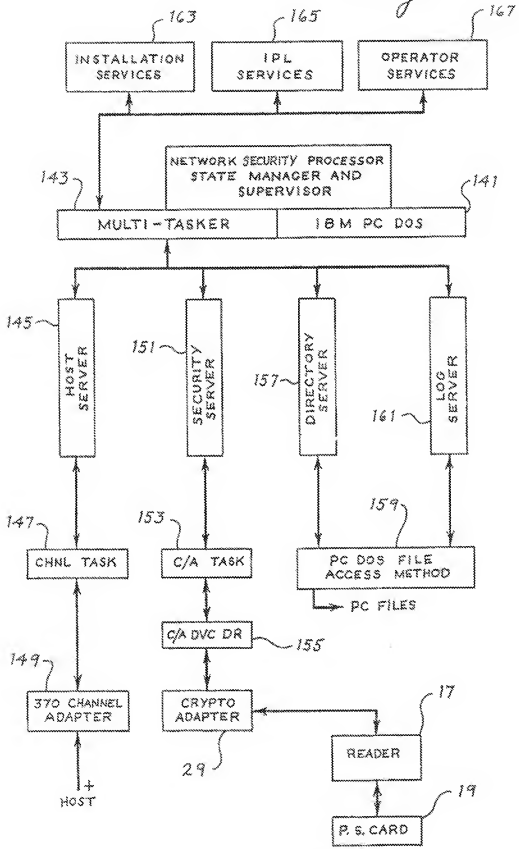

113
SECURITY UTILITIES
CUSTOMERS APPLICATIONS
SAPI STUB
111
WSSS SUPERVISOR - ROUTER
115
SECURITY SERVER
117
SERVER'S CALLER INTERFACE - SYSTEM UNIQUE
KEY MGMT
MAC VER
MAC GEN
123
127
125
ENCIPHER
DECIPHER
129
121
131
DEVICE DRIVER
119
61
CRYPTOGRAPHIC ADAPTER
READER
17
KEY STORAGE
29
SIGNATURE PEN
19

Fig. 6


163
165
167
INSTALLATION SERVICES
IPL SERVICES
OPERATOR SERVICES
NETWORK SECURITY PROCESSOR STATE MANAGER AND SUPERVISOR
143
141
MULTI-TASKER
IBM PC DOS
145
HOST SERVER
151
SECURITY SERVER
157
DIRECTORY SERVER
161
LOG SERVER
147
153
159
CHNL TASK
C/A TASK
PC DOS FILE ACCESS METHOD
PC FILES
C/A DVC DR
155
149
370 CHANNEL ADAPTER
CRYPTO ADAPTER
17
READER
29
19
P. S. CARD
HOST

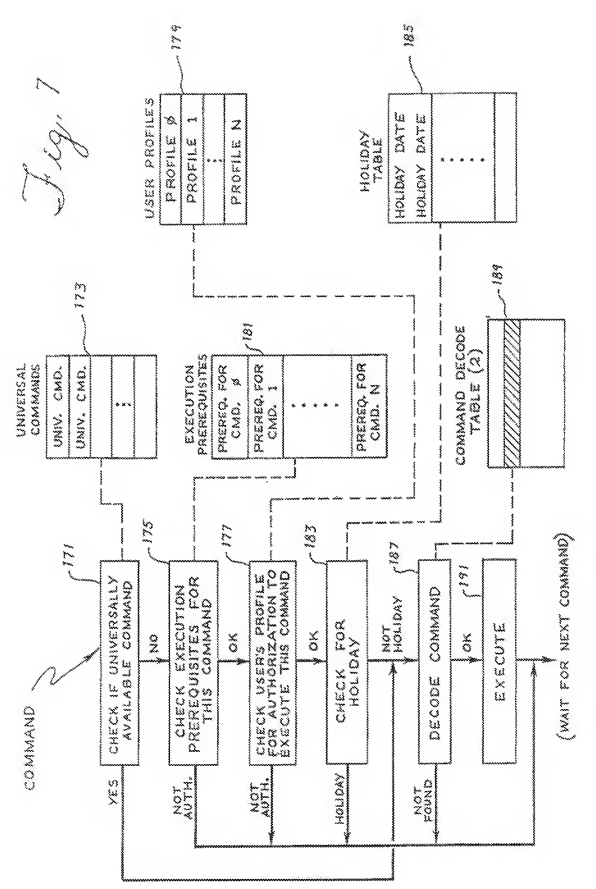

COMMAND
171
CHECK IF UNIVERSALLY AVAILABLE COMMAND
YES
NOT AUTH.
175
NO
CHECK EXECUTION PREREQUISITES FOR THIS COMMAND
177
OK
NOT AUTH.
CHECK USER'S PROFILE FOR AUTHORIZATION TO EXECUTE THIS COMMAND
183
OK
HOLIDAY
CHECK FOR HOLIDAY
187
NOT HOLIDAY
NOT FOUND
DECODE COMMAND
191
OK
EXECUTE
(WAIT FOR NEXT COMMAND)
UNIVERSAL COMMANDS
173
UNIV. CMD.
UNIV. CMD.
EXECUTION PREREQUISITES
181
PREREQ. FOR CMD. 0
PREREQ. FOR CMD. 1
PREREQ. FOR CMD. N
COMMAND DECODE TABLE (2)
189
USER PROFILES
179
PROFILE 0
PROFILE 1
PROFILE N
HOLIDAY TABLE
185
HOLIDAY DATE
HOLIDAY DATE

Fig. 7

Fig. 8

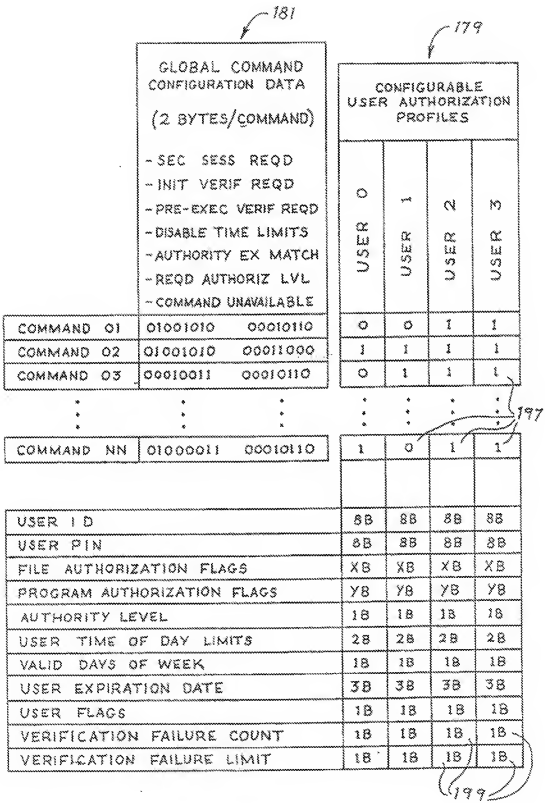

181
179
GLOBAL COMMAND CONFIGURATION DATA
(2 BYTES/COMMAND)
- SEC SESS REQD
- INIT VERIF REQD
- PRE-EXEC VERIF REQD
- DISABLE TIME LIMITS
- AUTHORITY EX MATCH
- REQD AUTHORIZ LVL
- COMMAND UNAVAILABLE
CONFIGURABLE USER AUTHORIZATION PROFILES
USER 0
USER 1
USER 2
USER 3
COMMAND 01 01001010 00010110 0 0 1 1
COMMAND 02 01001010 00011000 1 1 1 1
COMMAND 03 00010011 00010110 0 1 1 1
197
COMMAND NN 01000011 00010110 1 0 1 1
USER ID 8B 8B 8B 8B
USER PIN 8B 8B 8B 8B
FILE AUTHORIZATION FLAGS XB XB XB XB
PROGRAM AUTHORIZATION FLAGS YB YB YB YB
AUTHORITY LEVEL 1B 1B 1B 1B
USER TIME OF DAY LIMITS 2B 2B 2B 2B
VALID DAYS OF WEEK 1B 1B 1B 1B
USER EXPIRATION DATE 3B 3B 3B 3B
USER FLAGS 1B 1B 1B 1B
VERIFICATION FAILURE COUNT 1B 1B 1B 1B
VERIFICATION FAILURE LIMIT 1B 1B 1B 1B
199

Fig. 9A

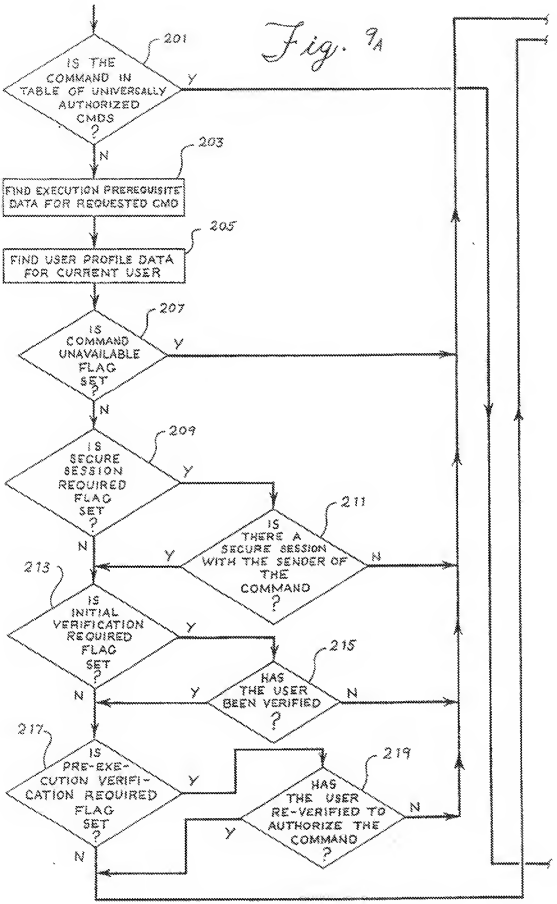
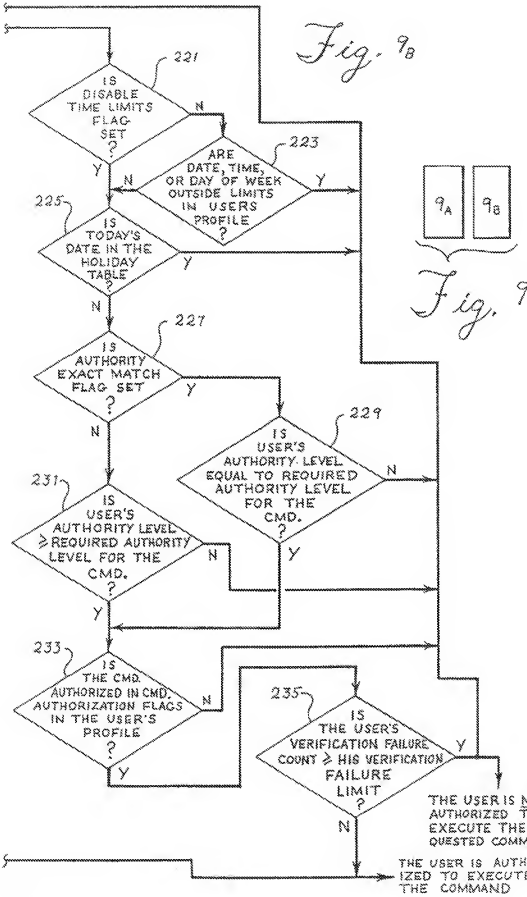

201
IS THE COMMAND IN TABLE OF UNIVERSALLY AUTHORIZED CMDS ?
Y
N
203
FIND EXECUTION PREREQUISITE DATA FOR REQUESTED CMD
205
FIND USER PROFILE DATA FOR CURRENT USER
207
IS COMMAND UNAVAILABLE FLAG SET ?
Y
N
209
IS SECURE SESSION REQUIRED FLAG SET ?
Y
N
211
IS THERE A SECURE SESSION WITH THE SENDER OF THE COMMAND ?
Y
N
213
IS INITIAL VERIFICATION REQUIRED FLAG SET ?
Y
N
215
HAS THE USER BEEN VERIFIED ?
Y
N
217
IS PRE-EXECUTION VERIFICATION REQUIRED FLAG SET ?
Y
N
219
HAS THE USER RE-VERIFIED TO AUTHORIZE THE COMMAND ?
Y
N

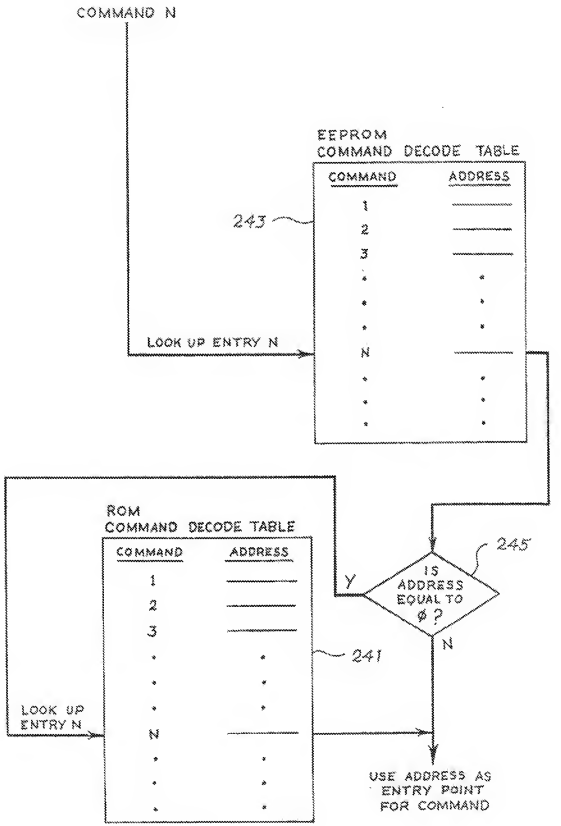
Fig. 9B


221
IS DISABLE TIME LIMITS FLAG SET ?
N
Y
223
ARE DATE, TIME, OR DAY OF WEEK OUTSIDE LIMITS IN USERS PROFILE ?
N
Y
225
IS TODAY'S DATE IN THE HOLIDAY TABLE ?
Y
N
227
IS AUTHORITY EXACT MATCH FLAG SET ?
Y
N
229
IS USER'S AUTHORITY LEVEL EQUAL TO REQUIRED AUTHORITY LEVEL FOR THE CMD. ?
N
Y
231
IS USER'S AUTHORITY LEVEL ≥ REQUIRED AUTHORITY LEVEL FOR THE CMD. ?
N
Y
233
IS THE CMD AUTHORIZED IN CMD. AUTHORIZATION FLAGS IN THE USER'S PROFILE ?
N
Y
235
IS THE USER'S VERIFICATION FAILURE COUNT ≥ HIS VERIFICATION FAILURE LIMIT ?
Y
N
THE USER IS NOT AUTHORIZED TO EXECUTE THE REQUESTED COMMAND
THE USER IS AUTHORIZED TO EXECUTE THE COMMAND

9A
9B

Fig. 9

Fig. 10


COMMAND N
EEPROM
COMMAND DECODE TABLE
COMMAND
ADDRESS
243
1
2
3
N
LOOK UP ENTRY N
ROM
COMMAND DECODE TABLE
COMMAND
ADDRESS
1
2
3
N
241
245
IS ADDRESS EQUAL TO ø?
Y
N
LOOK UP ENTRY N
USE ADDRESS AS ENTRY POINT FOR COMMAND

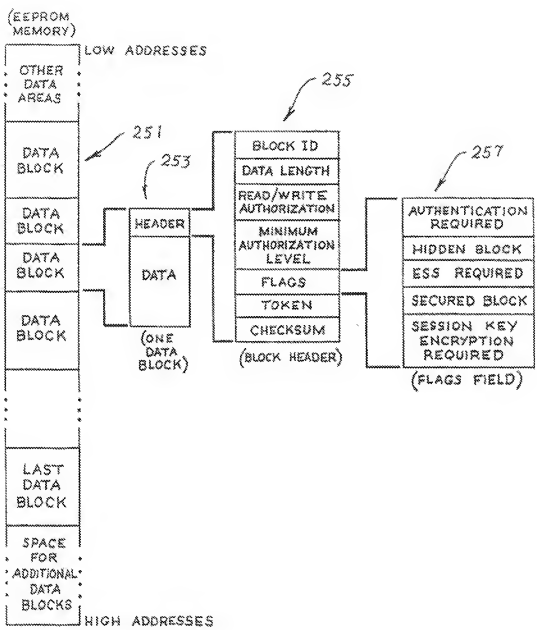

(EEPROM MEMORY)
LOW ADDRESSES
OTHER DATA AREAS
DATA BLOCK
DATA BLOCK
DATA BLOCK
DATA BLOCK
LAST DATA BLOCK
SPACE FOR ADDITIONAL DATA BLOCKS
HIGH ADDRESSES
251
253
255
257
HEADER
DATA
(ONE DATA BLOCK)
BLOCK ID
DATA LENGTH
READ/WRITE AUTHORIZATION
MINIMUM AUTHORIZATION LEVEL
FLAGS
TOKEN
CHECKSUM
(BLOCK HEADER)
AUTHENTICATION REQUIRED
HIDDEN BLOCK
ESS REQUIRED
SECURED BLOCK
SESSION KEY ENCRYPTION REQUIRED
(FLAGS FIELD)

Fig. 11

Command Class	Description or Examples
Status	Read device status
Control	Set/reset LED, generate tone, lock keypad, etc.
Cryptographic Functions	MAC, Encipher/Decipher, etc.
Key Management	Load keys, import keys, export keys, etc.
Data I/O	Allocate, read, write, delete, etc. for data blocks on the IC card
User Verification	Verify user's identity using PIN or signature verification, enroll user's signature, etc.
Secure Session	Establish secure cryptographic sessions between devices
Read Data	Read user data, configuration data, logs, etc.
Load Data	Load tables, configuration data, etc.
IML	Download new microcode to a device
Diagnostic	Communications tests, signature pen tests, etc.
Miscellaneous	Reset devices, set/read clock, etc.

Fig. 12

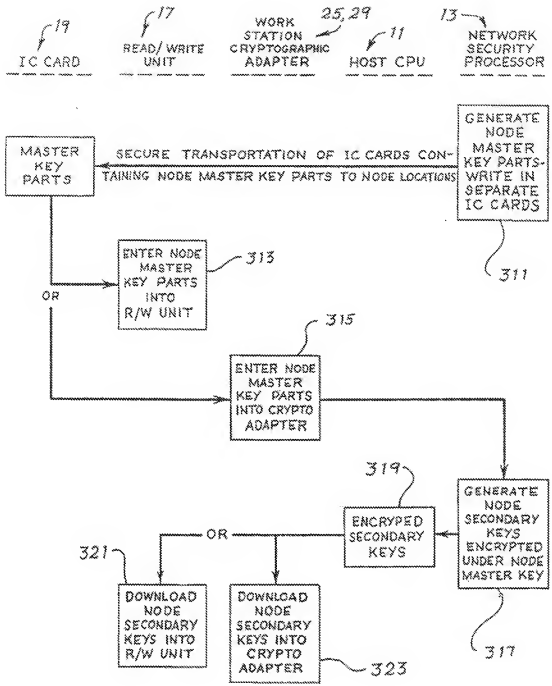

19
IC CARD
17
READ/ WRITE UNIT
WORK STATION CRYPTOGRAPHIC ADAPTER
25, 29
11
HOST CPU
13
NETWORK SECURITY PROCESSOR
MASTER KEY PARTS
SECURE TRANSPORTATION OF IC CARDS CONTAINING NODE MASTER KEY PARTS TO NODE LOCATIONS
GENERATE NODE MASTER KEY PARTS-WRITE IN SEPARATE IC CARDS
311
ENTER NODE MASTER KEY PARTS INTO R/W UNIT
313
OR
315
ENTER NODE MASTER KEY PARTS INTO CRYPTO ADAPTER
319
ENCRYPED SECONDARY KEYS
GENERATE NODE SECONDARY KEYS ENCRYPTED UNDER NODE MASTER KEY
317
OR
321
DOWNLOAD NODE SECONDARY KEYS INTO R/W UNIT
DOWNLOAD NODE SECONDARY KEYS INTO CRYPTO ADAPTER
323

Fig. 13

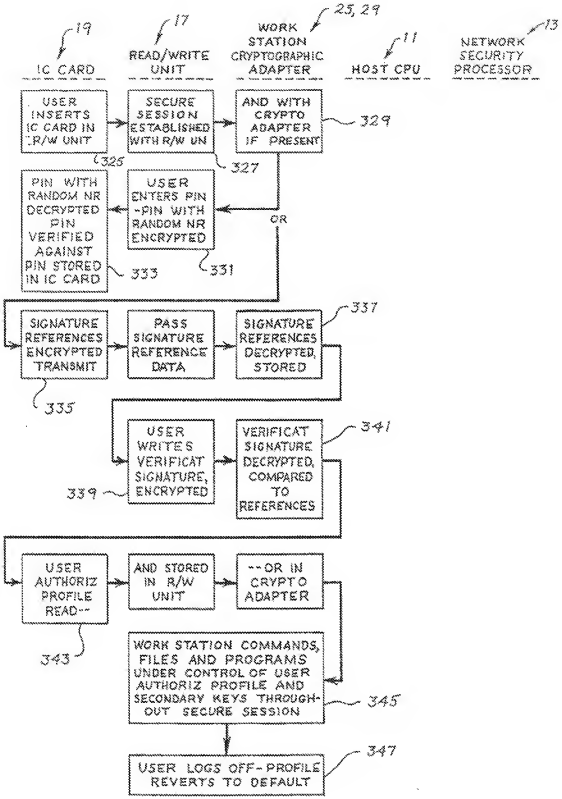

19
IC CARD
17
READ/WRITE UNIT
25, 29
WORK STATION CRYPTOGRAPHIC ADAPTER
11
HOST CPU
13
NETWORK SECURITY PROCESSOR
USER INSERTS IC CARD IN R/W UNIT
325
SECURE SESSION ESTABLISHED WITH R/W UN
327
AND WITH CRYPTO ADAPTER IF PRESENT
329
OR
USER ENTERS PIN -PIN WITH RANDOM NR ENCRYPTED
331
PIN WITH RANDOM NR DECRYPTED PIN VERIFIED AGAINST PIN STORED IN IC CARD
333
SIGNATURE REFERENCES ENCRYPTED TRANSMIT
335
PASS SIGNATURE REFERENCE DATA
SIGNATURE REFERENCES DECRYPTED, STORED
337
USER WRITES VERIFICAT SIGNATURE, ENCRYPTED
339
VERIFICAT SIGNATURE DECRYPTED, COMPARED TO REFERENCES
341
USER AUTHORIZ PROFILE READ--
343
AND STORED IN R/W UNIT
--OR IN CRYPTO ADAPTER
WORK STATION COMMANDS, FILES AND PROGRAMS UNDER CONTROL OF USER AUTHORIZ PROFILE AND SECONDARY KEYS THROUGHOUT SECURE SESSION
345
USER LOGS OFF- PROFILE REVERTS TO DEFAULT
347

Fig. 14

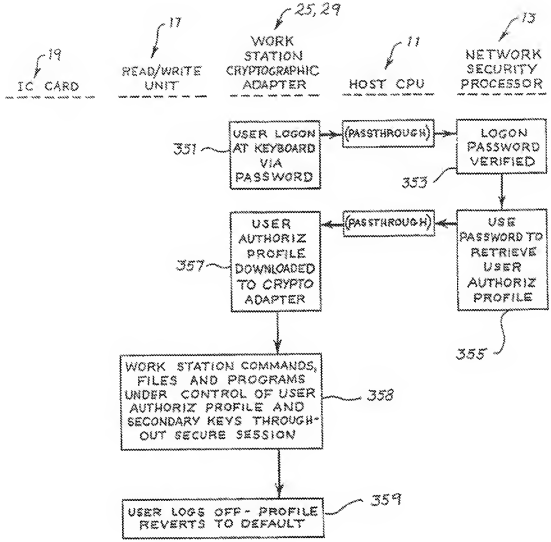

19
IC CARD
17
READ/WRITE UNIT
25, 29
WORK STATION CRYPTOGRAPHIC ADAPTER
11
HOST CPU
13
NETWORK SECURITY PROCESSOR
351
USER LOGON AT KEYBOARD VIA PASSWORD
PASSTHROUGH
LOGON PASSWORD VERIFIED
353
USE PASSWORD TO RETRIEVE USER AUTHORIZ PROFILE
PASSTHROUGH
USER AUTHORIZ PROFILE DOWNLOADED TO CRYPTO ADAPTER
357
355
WORK STATION COMMANDS, FILES AND PROGRAMS UNDER CONTROL OF USER AUTHORIZ PROFILE AND SECONDARY KEYS THROUGHOUT SECURE SESSION
358
USER LOGS OFF - PROFILE REVERTS TO DEFAULT
359

Fig. 15

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 510 224 A1**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **91106540.7**

(51) Int. Cl.5: **C25D 5/10**

(22) Date of filing: **23.04.91**

(43) Date of publication of application:
28.10.92 Bulletin 92/44

(84) Designated Contracting States:
DE FR GB

(71) Applicant: **NKK CORPORATION**
1-2 Marunouchi 1-chome, Chiyoda-ku
Tokyo(JP)

(72) Inventor: **Sagiyama, Masaru, c/o NKK Corporation**
1-2, 1-chome Marunouchi, Chiyoda-ku
Tokyo(JP)
Inventor: **Yoshida, Masafumi, c/o NKK Corporation**
1-2, 1-chome Marunouchi, Chiyoda-ku
Tokyo(JP)
Inventor: **Kawabe, Masaki, c/o NKK Corporation**
1-2, 1-chome Marunouchi, Chiyoda-ku
Tokyo(JP)
Inventor: **Ando, Satoru, c/o NKK Corporation**
1-2, 1-chome Marunouchi, Chiyoda-ku
Tokyo(JP)
Inventor: **Ono, Tadashi, c/o NKK Corporation**
1-2, 1-chome Marunouchi, Chiyoda-ku
Tokyo(JP)

(74) Representative: **Henkel, Feiler, Hänzel & Partner**
Möhlstrasse 37
W-8000 München 80(DE)

(54) Plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability.

(57) A plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability, which comprises: a steel sheet; a zinc plating layer formed on at least one surface of the steel sheet; and an electroplating layer formed on the zinc plating layer. The zinc plating layer has a plating weight of from 25 to 150 g/m^2 per surface of the steel sheet. The electroplating layer comprises at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, and the electroplating layer has a plating weight of from 1 to 10 g/m^2 per surface of the steel sheet.

EP 0 510 224 A1

REFERENCE TO PATENTS, APPLICATIONS AND PUBLICATIONS PERTINENT TO THE INVENTION

As far as we know, there is available the following prior art document pertinent to the present invention: "Plating & Surface Finishing", March 1989, pp. 62-69.

The contents of the prior art disclosed in the above-mentioned prior art document will be discussed hereafter under the heading of the "BACKGROUND OF THE INVENTION".

FIELD OF THE INVENTION

The present invention relates to a plated steel sheet having two plating layers and excellent in antifriction upon press-forming, corrosion resistance and painting adaptability.

BACKGROUND OF THE INVENTION

In general, the body of an automobile is exposed to a corrosive environment, and particularly to a severe corrosive environment in a coastal area or a cold area where an automobile tends to come into contact with a substance containing chlorine ions having a violent corrosivity.

A zinc plated steel sheet or a zinc alloy plated steel sheet is conventionally widely used as a steel sheet for an automobile body having an excellent corrosion resistance even in such a severe corrosive environment.

The conventional zinc plated steel sheet has however the following problems:

(1) The zinc plating layer of the zinc plated steel sheet, having a relatively low hardness, is deformed upon the press-forming of the zinc plated steel sheet, thus increasing a contact area between the zinc plating layer and the pressing portion of a press. In other words, the zinc plated steel sheet has a frictional coefficient higher than that of the other steel sheets such as a cold-rolled steel sheet or a zinc alloy plated steel sheet. When press-forming the zinc plated steel sheet, therefore, cracks may be produced in the zinc plating layer thereof. As is clear from the above description, the conventional zinc plated steel sheet is poor in antifriction (hereinafter referred to as the "problem 1"); and

(2) When applying an electropainting to the zinc plated steel sheet to form a painting film on the surface thereof, the high hydrogen overvoltage of the zinc plating layer causes the non-uniform production of a hydrogen gas during the electropainting, and the thus non-uniformly produced hydrogen gas which is confined in the painting film causes craters in the painting film, thus resulting in a poorer cratering resistance of the zinc plated steel sheet (hereinafter referred to as the "problem 2").

It is a conventional practice, as a means for solving the problem 1, to apply a high-viscosity lubricant oil onto the surface of the zinc plated steel sheet prior to press-forming same to improve antifriction of the zinc plated steel sheet.

Application of the high-viscosity lubricant oil onto the surface of the zinc plated steel sheet as described above poses however the following problems:

(a) The high-viscosity lubricant oil contaminates the working place; and

(b) It is necessary to remove the high-viscosity lubricant oil applied onto the surface of the zinc plated steel sheet prior to applying painting thereto. This removing operation is not however easy. Complete removal of the high-viscosity lubricant oil requires much time and labor.

With regard to the frictional coefficient of a zinc electroplated steel sheet, the "Plating & Surface Finishing", March 1989, pp. 62-69 teaches as follows (hereinafter referred to as the "prior art"):

(i) Application of a conventional anticorrosive oil onto the surface of the zinc electroplating layer having crystals oriented along the (0001) plane, leads to a relatively large frictional coefficient thereof of 0.19; and

(ii) Application of a conventional anticorrosive oil onto the surface of the zinc electroplating layer having crystals oriented along the $(10\bar{1}X)$ plane (where, X is 1, 2, 3 or 4), on the other hand, results in a small frictional coefficient thereof of 0.13.

Apart from the above-mentioned problems resulting from the application of the high-viscosity lubricant oil, the zinc electroplated steel sheet applied with the high-viscosity lubricant oil on the surface thereof has a small frictional coefficient of 0.11. If the orientation of the crystals of the zinc electroplating layer along the $(10\bar{1}X)$ plane (where, X is 1, 2, 3 or 4) as taught by the prior art can be maintained, an antifriction of the same order as in the application of the high-viscosity lubricant oil would be available by the application of the conventional anticorrosive oil which is easy to remove, onto the surface of the zinc electroplated steel sheet.

However, the crystal orientation of the zinc electroplating layer of the zinc electroplated steel sheet

depends upon electroplating conditions, and among others, upon an electric current density. As a result, it is inevitable to alter the plating conditions in response to the width, for example, of the steel sheet to be electroplated. In the manufacture of the zinc electroplated steel sheet in an industrial scale, it is practically impossible to maintain the orientation of the crystals of the zinc electroplating layer along the {101}X plane (where, X is 1, 2, 3 or 4).

A means to solve the problem 2 has not as yet been proposed.

Under such circumstances, there is a strong demand for the development of a plated steel sheet excellent in antifriction, corrosion resistance and painting adaptability, but such a plated steel sheet has not as yet been proposed.

SUMMARY OF THE INVENTION

An object of the present invention is therefore to provide a plated steel sheet having two electroplating layers and excellent in antifriction, corrosion resistance and painting adaptability.

In accordance with one of the features of the present invention, there is provided a plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability, characterized by comprising;

a steel sheet;

a zinc plating layer formed on at least one surface of said steel sheet, said zinc plating layer having a plating weight within a range of from 25 to 150 g/m^2 per surface of said steel sheet; and

an electroplating layer formed on said zinc plating layer, said electroplating layer comprising at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, and said electroplating layer having a plating weight within a range of from 1 to 10 g/m^2 per surface of said steel sheet.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a graph illustrating, for the plated steel sheet of the present invention, which has a zinc plating layer as a lower layer formed on the surface of the steel sheet and an electroplating layer as an upper layer formed on the zinc plating layer, the relationship between a frictional coefficient of the plated steel sheet and a plating weight of the electroplating layer as the upper layer; and

Fig. 2 is a schematic front view illustrating an apparatus for measuring a frictional coefficient.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

From the above-mentioned point of view, extensive studies were carried out to develop a plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability.

As a result, the following findings were obtained:

A plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability is available by the following steps:

(1) forming a zinc plating layer on at least one surface of a steel sheet,

(2) limiting a plating weight of the zinc plating layer within a range of from 25 to 150 g/m^2 per surface of the steel sheet;

(3) forming an electroplating layer comprising at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, on the zinc plating layer; and

(4) limiting a plating weight of the electroplating layer within a range of from 1 to 10 g/m^2 per surface of the steel sheet.

The present invention was made on the basis of the above-mentioned findings. The plated steel sheet of the present invention having two plating layers and excellent in antifriction, corrosion resistance and plating adaptability is described below with reference to the drawings.

The plated steel sheet of the present invention excellent in antifriction, corrosion resistance and painting adaptability comprises a steel sheet, a zinc plating layer as a lower layer formed on at least one surface of the steel sheet and an electroplating layer as an upper layer formed on the zinc plating layer.

The zinc plating layer as the lower layer formed on at least one surface of the steel sheet has a function of imparting an excellent corrosion resistance to the plated steel sheet.

The zinc plating layer is formed by either of the electroplating method and the dip-plating method, which are now widely diffused as an industrialized processes. When forming the zinc plating layer by the electroplating method, a zinc electroplating bath is selected from among a bath containing sulfate, a bath

containing chloride and a bath containing a mixture of sulfate and chloride. When forming the zinc plating layer by the dip-plating method, any one of the commonly utilized zinc dip-plating baths is used.

The plating weight of the zinc plating layer as the lower layer exerts an important effect on corrosion resistance, antifriction and press-formability of the plated steel sheet. With a plating weight of the zinc plating layer of under 25 g/m^2 per surface of the steel sheet, a desired corrosion resistance necessary for a rust-preventive steel sheet used as a material for an automobile body is not available. With a plating weight of the zinc plating layer of over 150 g/m^2 per surface of the steel sheet, on the other hand, when forming same by the electroplating method, zinc crystals of the zinc electroplating layer become coarser. As a result, a uniform electroplating layer as the upper layer cannot be formed on the zinc electroplating layer, thus making it impossible to obtain an effect of improving antifriction described later by the electroplating layer as the upper layer. In addition, with a plating weight of the zinc plating layer of over 150 g/m^2 per surface of the steel sheet, when forming the zinc plating layer by the dip-plating method, a zinc deposition in the width direction of the steel sheet shows a very non-uniform distribution, thus making it difficult to conduct the press-forming of the plated steel sheet. The plating weight of the zinc plating layer as the lower layer should therefore be limited within a range of from 25 to 150 g/m^2 per surface of the steel sheet.

The electroplating layer as the upper layer, which is formed on the zinc plating layer as the lower layer and comprises at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, has a function of imparting an excellent painting adaptability as typically represented by a high cratering resistance and an excellent antifriction. Since all the above-mentioned elements have a high melting point, it is difficult to form a plating layer of these elements as the upper layer on the zinc plating layer by the dip-plating method. The electroplating layer of these elements as the upper layer is therefore formed by the electroplating method with the use of an electroplating bath such as a bath containing sulfate, a bath containing chloride, a bath containing a mixture of sulfate and chloride, or a bath containing borofluoride.

Fig. 1 is a graph illustrating, for the plated steel sheet of the present invention, which has a zinc plating layer as a lower layer formed on the surface of the steel sheet and an electroplating layer as an upper layer formed on the zinc plating layer, the relationship between a frictional coefficient of the plated steel sheet and a plating weight of the electroplating layer as the upper layer. More particularly, a zinc electroplating layer as a lower layer having a plating weight of 60 g/m^2 per surface of a steel sheet was formed by the electroplating method on one surface of the steel sheet. Then, a nickel electroplating layer as an upper layer was formed on the zinc electroplating layer by the electroplating method while changing the plating weight of the nickel electroplating layer. A frictional coefficient was measured for the plated steel sheet having the thus formed two plating layers.

As is clear from Fig. 1, when the plating weight of the nickel electroplating layer as the upper layer is under 1 g/m^2 per surface of the steel sheet, the plated steel sheet has a high frictional coefficient. When the plating weight of the nickel electroplating layer as the upper layer is over 10 g/m^2 per surface of the steel sheet, on the other hand, the plated steel sheet has a constant frictional coefficient at a level of up to 0.1, which does not decrease to below this level. This applies, not only to the case where the above-mentioned nickel electroplating layer is formed as the upper layer, but also to the case where an electroplating layer comprising at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel is formed as the upper layer.

When the plating weight of the electroplating layer as the upper layer is under 1 g/m^2 per surface of the steel sheet, the frictional coefficient of the plated steel sheet is high as described above, so that antifriction of the plated steel sheet is deteriorated. When the plating weight of the electroplating layer as the upper layer is over 10 g/m^2 per surface of the steel sheet, on the other hand, the frictional coefficient of the plated steel sheet becomes constant at a level of up to 0.1.

As a result, not only the effect of the electroplating layer of improving antifriction of the plated steel sheet is saturated, but also corrosion resistance of the plated steel sheet becomes poorer. The plating weight of the electroplating layer as the upper layer, which comprises at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, should therefore be limited within a range of from 1 to 10 g/m^2 per surface of the steel sheet.

As described above, an excellent antifriction is imparted to the plated steel sheet, by forming the electroplating layer as the upper layer, which has a plating weight within a range of from 1 to 10 g/m^2 per surface of the steel sheet and comprises at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, on the zinc plating layer as the lower layer. The reason of this is estimated to be as follows:

The zinc plating layer of the zinc plated steel sheet has a relatively low hardness. Therefore, the zinc plating layer having a low hardness is deformed upon the press-forming of the zinc plated steel sheet. As a

result, a contact area between the zinc plating layer and the pressing portion of a press becomes larger, leading to a higher frictional coefficient of the zinc plated steel sheet. Antifriction of the zinc plated steel sheet therefore becomes lower. On the other hand, both a cold-rolled steel sheet and a zinc alloy plating layer of a zinc alloy plated steel sheet have a high hardness.

When the cold-rolled steel sheet or the zinc alloy plated steel sheet is press-formed, therefore, the surface of the cold-rolled steel sheet or the zinc alloy plating layer are hard to deform. As a result, a small contact area between the surface of the cold-rolled steel sheet or the zinc plating layer and the pressing portion of the press leads to a low frictional coefficient. The cold-rolled steel sheet or the zinc alloy plated steel sheet is therefore excellent in antifriction. The above description suggests that a higher hardness of the surface of a steel sheet provides a more excellent antifriction of the steel sheet.

In the present invention, the electroplating layer as the upper layer has a remarkably higher hardness than that of the zinc plating layer as the lower layer. When the plated steel sheet of the present invention is press-formed, therefore, the zinc plating layer as the lower layer deforms because of the low hardness thereof, and the electroplating layer as the upper layer is hard to deform because of the high hardness thereof. As a result, a small contact area between the surface of the electroplating layer and the pressing portion of the press leads to a low frictional coefficient of the plated steel sheet. The plated steel sheet of the present invention is therefore excellent in antifriction.

More particularly, when the plating weight of the electroplating layer as the upper layer is under 1 g/m^2 per surface of the steel sheet, most part of the surface of the zinc plating layer as the lower layer is exposed, and the contact area between the electroplated steel sheet and the pressing portion of the press becomes larger. This results in a high frictional coefficient of the plated steel sheet as in the above-mentioned zinc plated steel sheet. According as the plating weight of the electroplating layer as the upper layer increases from 1 g/m^2 per surface of the steel sheet, the electroplating layer as the upper layer more sufficiently covers the zinc plating layer as the lower layer, thus eliminating the exposed part of the zinc plating layer as the lower layer. This results in a smaller contact area between the plated steel sheet and the pressing portion of the press. As a result, the frictional coefficient of the plated steel sheet becomes very low. As described above, a plating weight of the electroplating layer as the upper layer of over 10 g/m^2 per surface of the steel sheet results in a constant frictional coefficient at a low level, which no longer decreases from this level.

In addition, the electroplating layer as the upper layer, which comprises at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, imparts an excellent painting adaptability, i.e., an excellent cratering resistance to the plated steel sheet.

More specifically, a hydrogen gas, if produced non-uniformly when forming a painting film by the electropainting on the surface of the plated steel sheet, is confined in the painting film, thus resulting in the occurrence of craters in the painting film.

However, since the electroplating layer as the upper layer of the plated steel sheet of the present invention, which comprises at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, has a low hydrogen overvoltage, a hydrogen gas is uniformly produced when forming a painting film by the electropainting. This results in a very rare occurrence of craters in the painting film, thus leading to an excellent painting adaptability, i.e., an excellent cratering resistance of the plated steel sheet.

Now, the plated steel sheet of the present invention having the two plating layers and excellent in antifriction, corrosion resistance and painting adaptability, is described further in detail by means of examples while comparing with examples for comparison.

EXAMPLES

Each of cold-rolled steel sheets having a thickness of 0.7 mm was subjected to a conventional degreasing treatment and a conventional pickling treatment to remove rust from the both surfaces thereof. Then, the steel sheet from the both surfaces of which rust was thus removed, was subjected to an electroplating under the conditions shown in Table 1 to form a zinc electroplating layer as a lower layer on each of the both surfaces of the steel sheet. In parallel with the above-mentioned electroplating, each of another cold-rolled steel sheets having a thickness of 0.7 mm, from the both surfaces of which rust was removed by the same method as described above, was subjected to a dip-plating under the following conditions to form a zinc dip-plating layer as a lower layer on each of the both surfaces of the steel sheet:

(a) Cold-rolled steel sheet : TH-F steel,

(b) Plating equipment: Continuous zinc dip-plating equipment having an annealing facility in the line,

(c) Plating bath: Zinc dip-plating bath containing aluminum in an amount of 0.14 wt.%,

(d) Plating bath temperature: 460 °C,

(e) Annealing temperature : 850 °C,

The plating weight was controlled by the gas squeezing method.

Then, each of the steel sheets having the zinc electroplating layer or the zinc dip-plating layer formed on each of the both surfaces thereof, was subjected to another electroplating under another conditions shown also in Table 1 to form an electroplating layer as an upper layer, which comprised at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, on the zinc electroplating layer or the zinc dip-plating layer. Thus, samples of the electroplated steel sheet within the scope of the present invention (hereinafter referred to as the "samples of the invention") Nos. 1 to 54 were prepared.

For each of the samples of the invention Nos. 1 to 54, the plating weight per surface of the steel sheet of the zinc electroplating layer or the zinc dip-plating layer, elements and the contents thereof of the electroplating layer, and the plating weight per surface of the steel sheet of the electroplating layer are shown also in Table 1.

Table 1 (2)

Example of the Invention No.	Plating conditions for forming zinc electroplating layer: Composition of plating bath (g/ℓ)																								Plating conditions for forming zinc electroplating layer								Electro-plating layer	Properties of Plated steel			
	Mn$_3$O$_4$ · 7H$_2$O	Al$_2$SO$_4$ · 18H$_2$O	Mg$_2$SO$_4$ · 7H$_2$O	FeSO$_4$ · 7H$_2$O	(NH$_4$)$_2$SO$_4$	CoSO$_4$ · 7H$_2$O	H$_3$BO$_3$	NiSO$_4$ · 6H$_2$O	CrO$_3$	H$_2$SO$_4$	MnSO$_4$ · 2H$_2$O	NH$_4$OCN	Cr$_2$ (SO$_4$)$_3$	(NH$_2$)$_2$CO	NaNO$_3$	ZnSO$_4$ · 7H$_2$O	Mg$_2$SO$_4$	CH$_3$COONa · 3H$_2$O	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm^2)	Electrodeposition time (sec.)	Plating weight (g/m^2)	Element and content thereof (wt.%)	Plating weight (g/m^2)	Electrolytic corrosion	Flaking resistance	Perforation resistance	Blister resistance
1	440	35	60	2.0	60	1.0	70	35.0	300	120	-	300	50	-	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	1.0	A	A	A	A
2	440	35	60	2.0	60	1.0	70	35.0	-	-	-	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.12	A	A	A	A
3	440	35	60	2.0	60	1.0	70	35.0	-	-	-	-	50	30	300	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
4	440	35	60	2.0	60	1.0	70	35.0	-	70	-	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
5	440	35	60	2.0	60	1.0	70	35.0	-	-	300	4.5	-	-	-	100	60	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.12	A	A	A	A
6	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.15	A	A	A	A
7	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
8	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
9	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
10	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
11	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
12	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
13	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
14	440	35	60	2.0	60	1.0	70	35.0	-	70	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
15	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
16	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
17	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A
18	440	35	60	2.0	60	1.0	70	35.0	300	120	150	300	50	30	-	-	-	-	-	2.0	60	3.0	50	1.0	70	35.0	60	0.11	A	A	A	A

Table 1 (2)

	No.	Plating conditions for forming zinc electroplating layer: Composition of plating bath (g/ℓ)								Plating conditions for forming electroplating layer: Composition of plating bath (g/ℓ)																				Zinc plating layer	Electro-plating layer		Properties of plated steel sheet			
		$ZnSO_4 \cdot 7H_2O$	Na_2SO_4	$Al_2(SO_4)_3 \cdot 18H_2O$	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm²)	Electroplating time (sec.)	$FeSO_4 \cdot 7H_2O$	$(NH_4)_2SO_4$	$CoSO_4 \cdot 7H_2O$	H_3BO_3	$NiSO_4 \cdot 6H_2O$	CrO_3	H_2SO_4	$MnSO_4 \cdot 2H_2O$	NH_4SCN	$Cr_2(SO_4)_3$	$(NH_2)_2CO$	$KMnO_4$	$SnSO_4 \cdot 7H_2O$	Na_2SO_4	$CH_3COONa \cdot 3H_2O$	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm²)	Electroplating time (sec.)	Plating weight (g/m²)	Element and content thereof (wt.%)	Plating weight (g/m²)	Frictional coefficient	Cratering resistance	Perforation resistance	Blister resistance
Sample of the invention	19	440	20	30	2.0	60	1.8	70	25.3	-	30	10	-	-	-	-	250	-	-	-	-	-	-	-	1.0	50	5.0	20	18.0	60	70Mn, 30Co	3	0.11	A	A	A
	20	440	20	30	2.0	60	1.8	70	25.2	-	30	-	20	-	-	-	250	-	-	-	-	-	-	-	1.0	50	5.0	30	14.3	60	70Mn, 30Ni	3	0.11	A	A	A
	21	440	20	30	2.0	60	1.8	70	25.3	-	-	-	-	-	20	0.2	-	-	-	-	100	-	-	-	1.0	50	5.0	30	7.8	60	70Mn, 30Cr	3	0.11	A	A	A
	22	440	20	30	2.0	60	1.8	70	25.3	50	-	-	-	-	-	-	-	-	200	50	-	-	-	-	2.0	50	3.0	50	2.9	60	60Cr, 40Fe	3	0.11	A	A	A
	23	440	20	30	2.0	60	1.8	70	25.3	-	-	30	30	-	-	-	-	-	200	-	-	-	-	-	2.0	50	3.0	50	3.5	60	60Cr, 40Co	3	0.13	A	A	A
	24	440	20	30	2.0	60	1.8	70	25.3	-	-	-	30	50	-	-	-	-	200	-	-	-	-	-	2.0	50	2.0	50	4.2	60	60Cr, 40Ni	3	0.11	A	A	A
	25	440	20	30	2.0	60	1.8	70	25.2	-	-	-	-	-	250	2	-	-	-	-	150	-	-	-	2.0	50	3.0	30	2.2	60	60Cr, 40Mn	3	0.11	A	A	A
	26	440	20	30	2.0	60	1.8	70	25.3	200	-	150	30	150	-	-	-	-	-	-	-	-	-	-	1.0	50	1.8	50	3.5	60	40Fe, 30Co, 30Ni	3	0.11	A	A	A
	27	440	20	30	2.0	60	1.8	70	25.3	50	30	-	-	-	-	-	200	-	50	-	-	-	-	-	2.0	50	1.8	50	4.2	60	80Fe, 10Mn, 10Cr	3	0.11	A	A	A
	28	440	20	30	2.0	60	1.8	70	25.2	250	-	-	20	250	-	-	-	-	100	-	-	-	-	-	1.0	50	1.0	70	5.9	60	80Fe, 15Ni, 5Cr	3	0.11	A	A	A
	29	440	20	30	2.0	60	1.8	70	25.3	-	-	-	-	100	-	-	200	-	100	-	-	-	30	20	2.0	50	1.8	20	23.0	60	80Ni, 15Mn, 5Cr	3	0.11	A	A	A
	30	440	20	30	2.0	60	1.8	70	25.2	10	-	-	-	-	-	-	250	-	10	-	-	-	30	20	2.0	50	1.8	30	2.2	60	80Mn, 15Fe, 5Cr	3	0.11	A	A	A
	31	440	20	30	2.0	60	1.8	70	25.3	-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.5	50	1.3	60	100Ni	3	0.10	A	A	A
	32	440	20	30	2.0	60	1.8	70	25.3	-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	4.2	60	100Ni	2	0.10	A	A	A
	33	440	20	30	2.0	60	1.8	70	25.2	-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	30	2.2	60	100Ni	10	0.09	A	A	A
	34	440	20	30	2.0	60	1.8	50	14.8	-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	2.5	25	100Ni	3	0.10	A	C	C
	35	440	20	30	2.0	60	1.8	70	13.7	-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	2.5	30	100Ni	3	0.11	A	C	B
	36	440	20	30	2.0	60	1.8	70	35.4	-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	30	4.2	60	100Ni	3	0.12	A	A	A

Table 1 (3)

	No.	Plating conditions for forming zinc electroplating layer: Composition of plating bath (g/ℓ) $ZnSO_4 \cdot 7H_2O$	Na_2SO_4	$Al_2(SO_4)_3 \cdot 16H_2O$	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm²)	Electroplating time (sec.)	Plating conditions for forming electroplating layer: Composition of plating bath (g/ℓ) $FeSO_4 \cdot 7H_2O$	$(NH_4)_2SO_4$	$CoSO_4 \cdot 7H_2O$	H_3BO_3	$NiSO_4 \cdot 6H_2O$	CrO_3	H_2SO_4	$MnSO_4 \cdot 5H_2O$	NH_4SCN	$Cr_2(SO_4)_3$	$(NH_2)_2CO$	$KMnO_4$	$SnSO_4 \cdot 7H_2O$	Na_2SO_4	$CH_3COONa \cdot 3H_2O$	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm²)	Electroplating time (sec.)	Zinc plating layer: Plating weight (g/m²)	Electroplating layer: Element and content thereof (wt.%)	Plating weight (g/m²)	Properties of plated steel sheet: Frictional coefficient	Cratering resistance	Perforation resistance	Blister resistance
Sample of the invention	37	refer to zinc dip-plating condition								-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	4.2	120	100Ni	5	0.14	A	A	A
	38	refer to zinc dip-plating condition								-	-	-	30	300	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	4.2	150	100Ni	5	0.14	A	A	A
	39	440	20	50	2.0	60	1.8	70	38.0	300	120	-	-	-	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	4.5	90	100Fe	5	0.13	A	A	A
	40	440	20	50	2.0	60	1.8	70	38.0	-	-	300	30	-	-	-	-	-	-	-	-	-	-	-	2.0	50	2.0	50	3.5	90	100Co	5	0.12	A	A	A
	41	440	20	50	2.0	60	1.8	70	38.0	-	75	-	-	-	-	-	100	60	-	-	-	-	-	-	2.0	50	5.1	10	60	90	100Mn	5	0.12	A	A	A
	42	440	20	50	2.0	60	1.8	70	38.0	-	-	-	-	-	250	2.5	-	-	-	-	-	-	-	-	2.0	50	1.5	50	100	90	100Cr	5	0.13	A	A	A
	43	440	20	50	2.0	60	1.8	70	38.0	250	120	50	-	-	-	-	-	-	-	-	-	-	-	-	2.0	50	1.8	50	4.4	90	80Fe, 20Co	5	0.12	A	A	A
	44	440	20	50	2.0	60	1.8	70	38.0	250	120	-	-	50	-	-	-	-	-	-	-	-	-	-	2.0	50	1.5	100	2.0	90	80Fe, 20Ni	5	0.13	A	A	A
	45	440	20	50	2.0	60	1.8	70	38.0	5	50	-	-	-	-	-	250	-	-	-	-	-	-	-	2.0	50	2.0	50	4.7	90	90Fe, 10Mn	5	0.11	A	A	A
	46	440	20	50	2.0	60	1.6	70	38.0	250	-	-	-	-	-	-	-	-	100	50	-	-	-	-	2.0	50	1.2	50	8.8	90	90Fe, 10Cr	5	0.11	A	A	A
	47	440	20	50	2.0	60	1.8	70	38.0	100	-	200	30	-	-	-	-	-	-	-	-	-	-	-	1.0	35	5.0	20	14.8	90	70Co, 30Fe	5	0.11	A	A	A
	48	440	20	50	3.0	60	1.8	70	38.0	100	-	-	50	400	-	-	-	-	-	-	-	-	-	-	1.0	35	5.0	20	10.5	90	80Ni, 20Fe	5	0.12	A	A	A
	49	440	20	50	2.0	60	1.8	70	38.0	-	150	-	-	80	-	-	250	-	-	-	-	-	-	-	1.0	50	3.0	20	15.8	90	80Ni, 20Mn	5	0.11	A	A	A
	50	440	20	50	2.0	60	1.8	70	38.0	-	-	-	30	100	-	-	-	-	200	-	-	-	-	-	2.0	50	3.0	50	8.8	90	80Ni, 20Cr	5	0.11	A	A	A
	51	440	20	50	2.0	60	1.6	70	38.0	-	50	-	20	-	-	-	250	-	-	-	-	-	-	-	1.0	50	5.5	30	24.0	90	70Mn, 30Ni	5	0.11	A	A	A
	52	440	20	50	2.0	60	1.8	70	38.0	200	-	150	30	150	-	-	-	-	-	-	-	-	-	-	1.0	50	1.8	50	4.7	90	40Fe, 30Co, 30Ni	5	0.11	A	A	A
	53	440	20	50	2.0	60	1.8	70	38.0	50	30	-	-	-	-	-	200	-	50	-	-	-	-	-	2.0	50	1.8	50	7.2	90	80Fe, 10Mn, 10Cr	5	0.12	A	A	A
	54	440	20	50	2.0	60	1.8	70	38.0	250	-	-	20	250	-	-	-	-	100	-	-	-	-	-	1.0	50	1.6	70	9.3	90	80Fe, 15Ni, 5Cr	5	0.11	A	A	A

For comparison purposes, each of steel sheets identical with those in the samples of the invention Nos. 1 to 54 (except for Nos. 37 and 38), was subjected to a conventional degreasing treatment and a conventional pickling treatment to remove rust from the both surfaces thereof. Then, the steel sheet from the both surfaces of which rust was thus removed, was subjected to an electroplating under the conditions shown in Table 2 to form a zinc electroplating layer as a lower layer on each of the both surfaces of the

steel sheet.

Then, each of the steel sheets having the zinc electroplating layer formed on each of the both surfaces thereof, was subjected to another electroplating under another conditions shown also in Table 2 to form an electroplating layer as an upper layer on the zinc electroplating layer. Thus, samples of the electroplated steel sheet outside the scope of the present invention (hereinafter referred to as the "samples for comparison") Nos. 1 to 7 were prepared. Each of the samples for comparison Nos. 1 and 7 had only a zinc electroplating layer as a single layer.

The plating weight per surface of the steel sheet of the zinc electroplating layer as the lower layer for each of the samples for comparison Nos. 1 to 7, as well as elements and the contents thereof of the electroplating layer as the upper layer, and the plating weight per surface of the steel sheet of the above-mentioned electroplating layer for each of the samples for comparison Nos. 2 to 6 are also shown in Table 2.

Table 2

No.	Plating conditions during formation of electroplating layer: Composition of plating bath (g/ℓ): $ZnSO_4 \cdot 7H_2O$	$NaZnO_4$	$Al_2(SO_4)_3 \cdot 18H_2O$	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm²)	Electroplating time (sec.)	Plating conditions during formation of electroplating layer: Composition of plating bath (g/ℓ): $FeSO_4 \cdot 7H_2O$	$(NH_4)_2SO_4$	$CoSO_4 \cdot 7H_2O$	$NiSO_4$	H_3BO_3	CrO_3	H_2SO_4	$MnSO_4 \cdot 7H_2O$	NH_4SCN	$CeCl_2(SO_4)_3$	$(NH_2)_2CO$	$KMnO_4$	$ZnSO_4 \cdot 7H_2O$	Na_2WO_4	$CH_3COOH \cdot 3H_2O$	Flow velocity of plating bath (m/sec.)	Bath temperature (°C)	pH value	Electric current density (A/dm²)	Electroplating time (sec.)	Plating weight (g/m²)	Elements contained in plating layer: Element and content thereof (wt.%)	Plating weight (g/m²)	Properties of plated steel sheet: Frictional coefficient	Coatering resistance	Refrigeration resistance	Blister resistance
Sample for comparison																																			
1	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
2	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
3	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
4	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
5	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
6	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
7	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
8	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
9	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
10	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
11	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
12	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
13	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
14	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
15	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
16	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
17	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
18	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
19	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
20	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
21	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
22	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
23	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
24	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
25	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
26	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
27	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
28	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
29	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
30	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
31	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
32	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
33	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
34	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
35	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
36	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
37	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
38	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
39	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
40	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
41	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
42	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
43	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
44	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
45	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
46	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
47	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
48	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
49	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
50	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
51	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
52	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
53	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
54	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
Comparison																																			
1	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
2	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
3	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
4	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
5	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
6	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A
7	340	40	60	0.5	60	1.8	70	25.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	A

Then, for each of the thus prepared samples of the invention Nos. 1 to 54 and the samples for comparison Nos. 1 to 7, antifriction, corrosion resistance and painting adaptability were investigated by means of performance tests as described below. The results of these tests are shown also in Tables 1 and 2.

(1) Antifriction test:

A mineral oil type anticorrosive oil for a steel sheet (product name: NOX RUST 530F40) manufactured by Parker Industries, Inc. was applied onto one surface of each of the samples of the invention Nos. 1 to 54 and the samples for comparison Nos. 1 to 6, and a high-viscosity lubricant oil (product name: FERROCOTE 61-MAL-HCL-1) manufactured by Nippon Quaker Chemical Co., Ltd. was applied onto one surface of the sample for comparison No. 7. The frictional coefficient of each sample on one surface of which the anticorrosive oil was applied and the frictional coefficient of the sample for comparison No. 7 on one surface of which the high-viscosity lubricant was applied, were measured with the use of an apparatus as shown in Fig. 2, thereby evaluating antifriction of each sample on the basis of the thus measured frictional coefficient.

The apparatus for measuring frictional coefficient of the sample comprised, as shown in Fig. 2, a rack 2; a supporting stand 5, provided on the rack 2 vertically movably along a plurality of guide rods 12 and 13 attached vertically to the rack 2, and having a plurality of rollers 6 on the upper end thereof; a supporting stand driving mechanism (not shown) for vertically moving the supporting stand 5; a first load cell 8, provided between the supporting stand 5 and the rack 2, for measuring the force applied vertically to the supporting stand 5; a pressing block 4 fitted to a frame 3 fixed to the rack 2 so as to project toward the supporting stand 5; a horizontally movable sliding table 7 mounted on the rollers 6 of the supporting stand 5 between the supporting stand 5 and the pressing block 4; a sliding table driving mechanism (not shown), provided on another rack 11, for horizontally moving the sliding table 7; and a second load cell 9, provided between an operating rod 10 connected to the sliding table driving mechanism and one end of the sliding table 7, for measuring the force applied horizontally to the sliding table 7.

By operating the supporting stand driving mechanism, the supporting stand 5 was moved upward to lift up the sliding table 7 on the upper surface of which a sample 1 was placed. Thus the upper surface of the sample 1 was pressed against the lower end of the pressing block 4, and the force N applied in the arrow A direction was measured by means of the first load cell 8. Then, by operating the sliding table driving mechanism, the sliding table 7 was horizontally moved in the arrow B direction, together with the sample 1 placed on the upper surface thereof, and the force F applied in the arrow B direction to the sliding table 7 was measured by means of the second load cell 9, at the moment when the sliding table 7 reached the moving speed of 1 m/minute. The ratio of the force F to the force N, i.e., the ratio F/N was determined, and the thus determined value was used as the value of frictional coefficient.

(2) Cratering resistance test:

Each of the samples having a width of 70 mm and a length of 150 mm was subjected to a dipping type phosphating for a steel sheet for automobile in a phosphating solution (product name: PBL 3080) manufactured by Nihon Perkerizing Co., Ltd., to form a phosphate film on the surface of the sample. Then, the sample was subjected to a cation type electropainting with the use of a paint (product name: ELECRON 9400) manufactured by Kansai Paint Co., Ltd. under the following conditions to form a painting film having a thickness of 20 µm on the phosphate film:

(a) Impressed voltage: 280 V,
(b) Area ratio between anode and cathode: 1:1,
(c) Electrification: instantaneous one,
(d) Electrifying time: 3 minutes.

Cratering resistance was evaluated by means of the number of craters produced in the painting film during the formation of the painting film as described above. The criteria for evaluation were as follows:

A: up to ten craters produced within a circle of a diameter of 40 mm at the center of the sample,
B: from 11 to 100 such craters,
C: at least 101 such craters.

(3) Corrosion resistance test:

Corrosion resistance was evaluated by means of perforation resistance and blister resistance as follows:

(a) Perforation resistance:

As in the cratering resistance test as described above, a phosphate film was formed on the surface of each of the samples, and a painting film having a thickness of 20 µm was formed on the phosphate film by means of the electropainting. Then, a notch was provided in the thus formed painting film. Each of the

samples having the thus notched painting film was then subjected to 60 cycles of corrosion tests, each cycle comprising a salt water spray, a drying, a dipping into salt water, and a drying for 24 hours. Then, the painting film and corrosion products produced during the corrosion test, were removed from each sample thus subjected to the 60 cycles of corrosion tests, and the maximum corrosion depth produced in the steel sheet was measured. Perforation resistance was evaluated by means of the thus measured maximum corrosion depth. The criteria for evaluation were as follows:

A: A maximum corrosion depth of under 0.1 mm,
B: A maximum corrosion depth within a range of from 0.1 to under 0.2 mm,
C: A maximum corrosion depth within a range of from 0.2 to under 0.4 mm, and
D: A maximum corrosion depth of at least 0.4 mm.

(b) Blister resistance:

As in the cratering resistance test as described above, a phosphate film was formed on the surface of each of the samples, and a lower painting film having a thickness of 10 µm was formed on the phosphate film by means of the electropainting. Then, an intermediate painting film having a thickness of 35 µm and an upper painting film having a thickness of 35 µm were formed on the thus formed lower painting film. Then, a notch was provided on the thus formed triple-layer painting film. A salt spray test was carried out on each of the samples having the thus notched triple-layer painting film. More specifically, each sample was exposed to the open air for a period of one year, during which salt water having a sodium chloride content of 5 wt.% was sprayed over the sample at a rate of twice a week. Then, the maximum blister width of the painting film was measured on one side of the notch on the sample after the salt spray test, and blister resistance was evaluated by means of the thus measured maximum blister width of the painting film. The criteria for evaluation were as follows:

A: A maximum blister width of under 1 mm;
B: A maximum blister width within a range of from 1 mm to under 2 mm;
C: A maximum blister width within a range of from 2 mm to under 2.5 mm; and
D: A maximum blister width of at least 2.5 mm.

As is clear from Table 1, all the samples of the invention Nos. 1 to 54 had a frictional coefficient of up to 0.16, and were therefore excellent in antifriction.

All the samples of the invention Nos. 1 to 54 were excellent in painting adaptability as typically represented by a high cratering resistance with the number of craters of up to 10 produced within a circle of a diameter of 40 mm at the center of the sample, as evaluated by A.

In terms of perforation resistance and blister resistance, which represented corrosion resistance, although the samples of the invention Nos. 34 and 35 were slightly poor, all the other samples of the invention were excellent. Each of the samples of the invention Nos. 34 and 35 was slightly inferior in corrosion resistance to each of the samples of the invention Nos. 1 to 33 and 36 to 54 because the plating weight of the zinc electroplating layer of each of the samples of the invention Nos. 34 and 35 was smaller than that of each of the samples of the invention Nos. 1 to 33 and 36 to 54.

As is evident from the above description, all the samples of the invention Nos. 1 to 54 were excellent in antifriction, corrosion resistance and painting adaptability.

As is clear from Table 2, in contrast, none of the samples for comparison Nos. 1 to 7 satisfied simultaneously the following three favorable merits possessed by each of the samples of the invention Nos. 1 to 54:

(i) a frictional coefficient of up to 0.18 in the antifriction test;
(ii) a maximum blister width of under 2.5 mm and a maximum corrosion depth of under 0.4 mm in the corrosion test; and
(iii) a number of craters representing painting adaptability of up to 10 in the cratering resistance test.

More particularly, the sample for comparison No. 1 having the zinc electroplating layer as the single layer and applied with the anticorrosive oil for a steel sheet on the surface thereof had a large frictional coefficient of 0.3. The sample for comparison No. 2 having a low plating weight of the nickel electroplating layer as the upper layer outside the scope of the present invention had a large frictional coefficient of 0.28. The sample for comparison No. 3 having a high plating weight of the nickel electroplating layer as the upper layer outside the scope of the present invention was poor in perforation resistance and blister resistance. The sample for comparison No. 4 having the zinc-iron alloy electroplating layer as the upper layer and the sample for comparison No. 5 having the zinc-nickel alloy electroplating layer as the upper layer, which were outside the scope of the present invention in that the electroplating layer as the upper layer contained zinc, were poor in cratering resistance. The sample for comparison No. 6 having a low plating weight of the zinc

electroplating layer as the lower layer outside the scope of the present invention was poor in perforation resistance and blister resistance.

Furthermore, a high-viscosity lubricant oil (product name: FERROCOTE 61-MAL-HCL-1) manufactured by Nippon Quaker Chemical Co., Ltd. was applied onto the zinc electroplating layer as the single layer of the sample for comparison No. 7, and an antifriction test as described above was effected on the sample for comparison No. 7 applied with the high-viscosity lubricant oil on the zinc electroplating layer thereof. The above-mentioned sample for comparison No. 7 had a frictional coefficient of 0.11. This revealed that the samples of the invention Nos. 1 to 54 applied with the easily removable anticorrosive oil had substantially the same antifriction as that of the sample for comparison No. 7 applied with the high-viscosity lubricant oil which was very difficult to remove.

According to the present invention, as described above in detail, it is possible to provide a plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability, thus providing industrially useful effects.

Claims

1. A plated steel sheet having two plating layers and excellent in antifriction, corrosion resistance and painting adaptability, characterized by comprising:

   a steel sheet;

   a zinc plating layer formed on at least one surface of said steel sheet, said zinc plating layer having a plating weight within a range of from 25 to 150 g/m^2 per surface of said steel sheet; and

   an electroplating layer formed on said zinc plating layer, said electroplating layer comprising at least one element selected from the group consisting of chromium, manganese, iron, cobalt and nickel, and said electroplating layer having a plating weight within a range of from 1 to 10 g/m^2 per surface of said steel sheet.

2. A plated steel sheet as claimed in Claim 1, wherein:
   said zinc plating layer is an zinc electroplating layer.

3. A plated steel sheet as claimed in Claim 1, wherein:
   said zinc plating layer is a zinc dip-plating layer.

4. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
   said electroplating layer comprises chromium.

5. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
   said electroplating layer comprises manganese.

6. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
   said electroplating layer comprises iron.

7. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
   said electroplating layer comprises cobalt.

8. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
   said electroplating layer comprises nickel.

9. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
   said electroplating layer comprises iron and cobalt.

10. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
    said electroplating layer comprises iron and nickel.

11. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein:
    said electroplating layer comprises iron and manganese.

12. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises iron and chromium.

13. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises cobalt and nickel.

14. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises cobalt and manganese.

15. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises cobalt and chromium.

16. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises nickel and manganese.

17. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises nickel and chromium.

18. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises manganese and chromium.

19. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises iron, cobalt and nickel.

20. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises iron, manganese and chromium.

21. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises iron, nickel and chromium.

22. A plated steel sheet as claimed in any one of Claims 1 to 3, wherein said electroplating layer comprises nickel, manganese and chromium.

FIG. 1

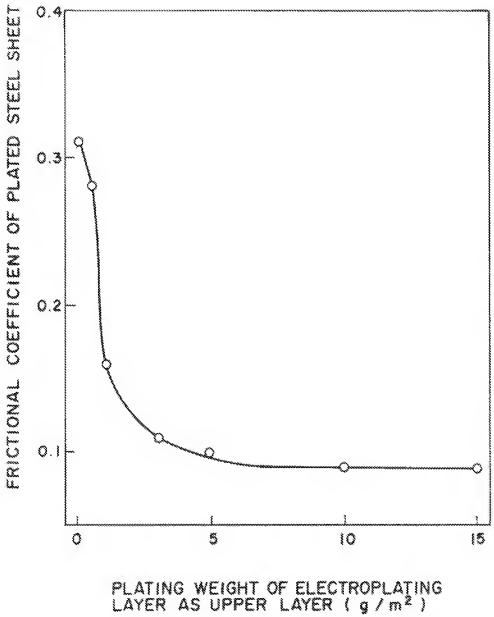
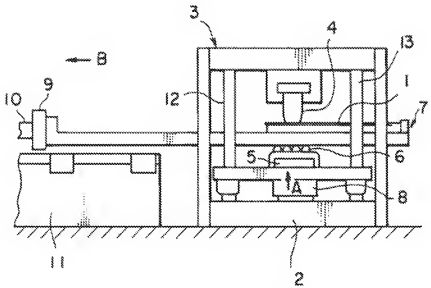

FRICTIONAL COEFFICIENT OF PLATED STEEL SHEET
0.4
0.3
0.2
0.1
0
5
10
15
PLATING WEIGHT OF ELECTROPLATING LAYER AS UPPER LAYER (g / m²)

FIG. 2



European Patent Office

EUROPEAN SEARCH REPORT

Application Number

EP 91 10 6540

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int. Cl.5)
X	PATENT ABSTRACTS OF JAPAN vol. 8, no. 114 (C-225)(1551) 26 May 1984 & JP-A-59 025 992 (KAWASAKI SEITETSU KK) 10 February 1984 * abstract *	1,2,15	C25D5/10
X	GB-A-2 161 499 (PHENIX WORKS) * page 2, line 30 - line 65 * * page 2; claim 3 *	1,3,4,5,6,7,8	
X	US-A-3 323 881 (NELSON) * column 3, line 30 - line 48; claims 3,4 *	1,2,3,4	
			TECHNICAL FIELDS SEARCHED (Int. Cl.5)
			C25D

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	18 DECEMBER 1991	NGUYEN THE NGHIEP N.

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P0401)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 849 680 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
24.06.1998 Bulletin 1998/26

(51) Int. Cl.[6]: G06F 12/14, G06F 1/00

(21) Application number: 97122255.9

(22) Date of filing: 17.12.1997

(84) Designated Contracting States
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 18.12.1996 US 769603

(71) Applicant:
SUN MICROSYSTEMS, INC.
Palo Alto, California 94303 (US)

(72) Inventor: Winiger, Gary W.
Mountain View, California (US)

(74) Representative:
Kahler, Kurt, Dipl.-Ing. et al
Patentanwälte
Kahler, Käck, Fiener et col.,
Vorderer Anger 268
86899 Landsberg/Lech (DE)

(54) Multilevel security port methods, apparatuses, and computer program products

(57) A multilevel port system on a computer operating under a multilevel operating system to permit contemporaneously opening a plurality of sockets having the same port number while meeting the requirements of an appropriate security policy, thus allowing third party applications to run as if they were unimpeded by the security policy, and methods thereby. The computer system having an operating system adhering to an access control security mechanism. Such systems include government systems wherein a hierarchy of security classification levels are defined (e.g., top secret, secret, classified, unclassified), and commercial systems. Sensitivity labels pursuant to an access control security mechanism include at least hierarchical security classifications, and may include non-hierarchical categories or compartments which represent distinct areas of information in a system. A port is characterized by a port number and a sensitivity label thus permitting opening a plurality of ports having identical port numbers and unique sensitivity labels.

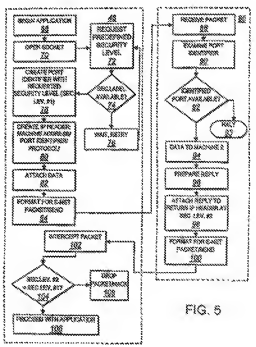


FIG. 5

EP 0 849 680 A2

Description

TECHNICAL FIELD

The present invention relates to multilevel port methods, apparatuses, and computer program products operable in computer systems, and more particularly, to multilevel port systems operable in multilevel operating systems utilizing multilevel multiple security levels.

BACKGROUND

Secure computer systems restrict information from unauthorized disclosure. Government secrecy systems ensure that users access only permitted information in accordance with predetermined security clearances. Other secure environments protect selected private information including payroll data and other sensitive company data including internal memoranda and competitive strategy documents.

To establish computer security for government or company systems, a security policy is adopted. The security policy establishes rules for managing, protecting and distributing sensitive information. A security policy is typically stated in terms of subject and objects. Subjects are active within a selected system and include users, processes, and programs, for example. Objects are the recipients of subject action, such as files, directories, devices, sockets, and windows. A security policy may set rules to determine whether a subject user has access to a particular object such as a file.

One well-known security system developed by David Bell and Leonard LaPadula in 1973 describes a multilevel secure computer system having access rules depending upon the security clearances of messaging processes. Security systems based upon access rules rely upon reference monitors which enforce authorized access relationships between subjects and objects of a system. A security kernel concept developed by Roger Shell in 1972 implements the reference monitor notion that all system activity is supervised in accordance with the system's security policy. The kernel accordingly mediates. A "trusted system" has sufficient hardware and software integrity to allow its use to simultaneously process a range of sensitive unclassified or classified information for a diverse set of users without violating access privileges.

Networks require that the security mechanism of a trusted system be able to control communication with the trusted systems. Previously, a network administrator typically had tight control over system connections with other systems. However, with the proliferation of interconnected networks and easy remote access and resource sharing, systems often cannot identify or trust the entire network.

Strategies for establishing security in network environments require labeling data with predetermined security attributes or sensitivity labels, information labels. This enables recognition of data sensitivity at other systems of a network. Because different networks support different security policies, these labels are not necessarily in the same format. In certain secure networks, each system may have a different kind of label. A user sensitivity label specifies the sensitivity level, or level of trust, associated with that user. A file's sensitivity label similarly specifies the level of trust that a user must have to be able to access the particular file. Mandatory access controls use sensitivity labels to determine who can access what information in a system. Together, labeling and mandatory access control implement a multilevel security policy - a policy for handling multiple information classifications at a number of different security levels within a single computer system.

Under mandatory access control, every subject and object in a system supporting mandatory access controls has a sensitivity label associated with it. A sensitivity label generally includes a classification and a set of categories or compartments. The classification system is typically hierarchical, including in a military security model, for example, multiple distinct levels, such as top secret, secret, confidential and classified. In a company environment, other classifications may be followed including labels such as company confidential, or company private.

Typically, for a subject to read an object, the subject's sensitivity level must dominate the object's sensitivity level. A subject's sensitivity label dominates the object's sensitivity label if the subject's classification is equal to or exceeds the classification of the object. Similarly, in order to write an object, the object's sensitivity level must dominate the subject's sensitivity level. In order for a subject to write to an object, the subject's sensitivity level must be equal to or less than the sensitivity level of the object or file. Consequently, in a current mandatory access system, in order for a subject to freely read and write to and from an object, both the subject and the object must have the same classification label. This is the fundamental rule by which an access control system works, and by which two-way communication may take place between trusted computer systems.

In current networked multilevel trusted systems, third-party applications have only limited support for operating effectively. In particular, when multiple processes having different sensitivity labels attempt to access the same object or resource, despite differences in security level, the operation may block. In the prior art diagram of Figure 1, an application runs on a trusted system and attempts to access a resource (i.e., a file, an application, or a database) either on the same system or on another system in a network. For success, the security levels of resource and subject must necessarily be the same in order to permit two-way communication according to the applicable access control security mechanism.

In multilevel trusted systems of the prior art as shown diagrammatically in Figure 1, access to a resource or a service (object) by a process (subject) running at a particular sensitivity level is restricted to objects in memory having the same sensitivity level as the requesting process, as mandated by the access control mechanism. Consequently, two-way communication is precluded where the subject and the object have different sensitivity labels. Once a requested application, service or resource is instantiated in computer memory, a sensitivity label is associated with the process, service, or resource, and access by other processes running applications which also desire to access the resource, but which have a different clearance, is denied.

Another technical problem arises, however, in the prior art system of Figure 2 described below when a port on a receiving system remains open for a substantial period of time at a particular security classification, clearance level, or sensitivity label. This prevents users and systems having different clearances from accessing the same resource, when a port has already been opened and remains open under a different clearance. Since a port number is unique to a resource or third party system being accessed, the unavailability of that particular port effectively precludes other users or systems with different clearances from accessing the third party resource. This effectively renders the resource unavailable to applications operating at different security levels.

Accordingly, there is a need for systems and methods providing access to resources operating at multiple security levels. Such systems and methods must be transparent to processes having different security classification levels.

An additional problem with current multilevel trusted systems is security violations from interlevel signal channel communications between associated system ports or covert channels. A covert channel is an information path that is not ordinarily used for communication in a system and thus is not protected by the system's normal security mechanisms. Thus, there is a secret way to communicate information to another person or program in violation of security protocol. The covert channels convey information by changes in data attributes or by changes in system performance or timing. By monitoring attribute changes for stored data and system timing, confidential information may be inferred. Data characteristics such as message length, frequency, and destination may be protected from analysis of data traffic by an intruder or from a user having a lower classification on the same system, with techniques such as covert channel analysis, padding messages to disguise their actual characteristics, or by sending noise or spurious messages. However, such measures do not guarantee data security.

Accordingly, there is a need for systems and methods to prevent data access in violation of security protocol to ports having a dominant classification in a multisecurity level computer system. Such systems and methods must secure access to the dominant port to protect attribute information from compromise to an intruder.

SUMMARY OF THE INVENTION

The invention is defined in claims 1, 2, 5, 7 and 8, respectively.

According to the present invention, multilevel trusted systems associate multiple port endpoints with a single identifier code indication or name. Use of a single identification to associate multiple port endpoints enables provision of a security check which halts inter-endpoint communication when the endpoints are further associated with a common identifier code indication. This is beneficial because security breaches caused by interlevel communication are diminished.

According to the present invention, use privileges for third-party communication at a selected network level are affirmatively granted at multiple specified levels. This is beneficial as it permits direct and unmodified application operation at desired multiple levels, permitting multilevel trusted system operation without applications software modification.

According to the present invention, a computer system comprises a machine-readable program storage device embodying a program of instructions executable by the machine to perform method steps in a multilevel trusted system for establishing a multilevel port to enable multiple, substantially concurrent resource accessing.

According to the present invention, a computer system comprises an operating system kernel supporting a multilevel access control security mechanism for creating an object access packet comprising an internet protocol (IP) header including a destination socket having a machine address and a unique port identifier, a port identifier comprising a port number specifying a resource or object, and a sensitivity label for an access control security protocol. According to the present invention, a plurality of processes are created on a destination system for a single selected port number at a selected unique sensitivity label, permitting resource and object access by multiple users in a multilevel access control system to a selected port according to a selected security policy.

According to the method of this invention, machine readable code opens multiple instances of a selected application, both instances having the same port address and a separate sensitivity label.

According to the present invention, multiple network endpoints having the same port number but separate security classification labels are established, permitting contemporaneous process port access according to a common port number while still adhering to the system security policy. As many ports may be open with the

same port number as there are different security classifications used by the system access control security protocol.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of a multilevel trusted system having a plurality of ports and endpoints at predetermined security levels, according to the prior art;

Figure 2 is a flow diagram of a multilevel trusted system according to the prior art, in which a datagram or message packet is communicated between a source system and a destination system;

Figure 3 is a block diagram of a security system according to the prior art;

Figure 4 is a diagram of an internet system according to the present invention;

Figure 5 is a flow diagram of multilevel trusted system operation according to the present invention; and

Figure 6 is a diagram of a multilevel trusted system processing a communications packet according to the present invention.

DETAILED DESCRIPTION OF A BEST MODE OF THE INVENTION

Figure 2 is a flow diagram of a prior art system employing access control security mechanisms. Third party applications require a license verification from a remote third party computer system. Alternatively, license verification may be an object in a process operating on the same system as the process in which the application is running. Once an application is instantiated on a first operating system, it may determine that communication with an object process is required. The kernel on the first system accordingly creates 6 a socket, and constructs 8 a communications packet, including an appropriate header, a machine address, a port number, and a protocol identifier, attaches 10 a data and a sensitivity label continuing the clearance of the process under which the application is running, and transmits 12 through socket a data packet over a selected electronic communications medium.

An internet protocol (IP) header typically contains source system information for the system originating communication and information regarding the destination system. This information includes machine numbers of the source and the destination computers, the port numbers or addresses identifying applicable applications and services provided, and the protocol (e.g., TCP/IP, or UDP/IP) by which the two computers will communicate. Port numbers or addresses identify application or subject running on a client computer, and the application object or resource to be accessed on a destination machine such as a license verification program on a remote machine 13 or server.

During network communication, an IP header and data are electronically communicated 14 from the source system, through a socket endpoint for receipt 8 by a destination server. The destination kernel determines whether a requested port is available 20. If the port is available (i.e., not yet opened), the requested port opens 22 at a clearance level associated with the sensitivity label of the incoming communication. If the requested port number is in use, the request is dropped 32, possibly with a negative acknowledgment (NACK) being returned to the source server. The same classification level is required for two-way communication between a source system and a destination system under an access control security mechanism.

If a request is processed, the destination system opens 22 a port and prepares 16 a reply 16 and an IP header for the reply. An IP sensitivity label for the process under which the object application is running is additionally attached 28 to the reply. Under mandatory access control, the sensitivity label must contain the same security classification of the request of the originating system. The reply packet is further sent 26 to the originating server, where the packet is trapped 29 by the source kernel and inspected 30 pursuant to the security protocol for that system. If the reply packet is not provided at the same security level as the original request, the packet is dropped 32. Otherwise, the packet is passed-on 34 to the requesting application.

Figure 3 shows a multilevel trusted system according to the prior art, including first through fourth instances of the same selected application 40 running concurrently. The application instances of a running application are respective processes 42a - 42d. Each of processes 42a - 42d is assigned a particular security classification, and each process handles communication between application 40 and kernel 44. The assigned security classification, may be a predetermined clearance level based upon the identity of a user or a user category, or a type of application, for example. Kernel 44 controls input output functions, memory, processes, and operational aspects of running application 40. Kernel 44 mediates relationships 46 between processes of application 40 and selected resources 48, such as objects, services, and external application connecting to the processes of application 40. Kernel 44 includes a security process 50 ensuring that each process of application 40 communicates only with resources having a security classification consistent with a predetermined security policy. According to a mandatory access control (MAC) system, for example, security process 50 ensures that processes 42a - 42d only communicate with resources 48 at the same security classification as the corresponding process of application 40. All MAC objects are accordingly labeled with a security label which is used for communications packets traveling between the application process and the resource with which it has message traffic.

Figure 4 shows a multiuser, multilevel source

trusted computer system 50 according to the present invention, which is networked to a second computer system 54 through a communication network 55, such as the Internet. In a typical configuration, several users are networked into a server. Source trusted computer system 50 includes a network including a plurality of user workstations 56a-56, a server 58, and a gateway server 60, which may be employed as a firewall to prevent unauthorized access to source trusted computer system 50. The gateway server 60 includes a memory 61 for storing a kernel (not shown). The second computer system 54 includes a memory 62 for storing a kernel. For incoming messages, a security inspection is performed on incoming packets by the kernel (not shown) of gateway server 60. A received packet is passed into source trusted computer system 50 only after it has been determined that the packet has satisfied the security protocols of the source trusted computer system 50. In a multilevel trusted system using a mandatory access control security protocol, for example, the kernel of source trusted computer system 50 ensures that the sensitivity label of an incoming communications packet is the same as or higher than the sensitivity label of the destination process or port destination of computer system 54 to which the packet is addressed. If the packet security classification is not the same as or higher than the security classification destination port, then the packet is discarded from further processing. Message packets are sent through a modem 64 or a network interface card (not shown) over a selected transmission medium 62 formed of a copper wire, a fiber optic link, a microwave line, or a radio broadcast transmission link. The selected link with destination computer system 54 may be directly through a LAN connection, a direct phone link, or indirectly such as through the Internet. Upon reaching the destination computer system server, the message packet is intercepted by the server kernel (not shown). Should the destination server employ an OSI interface, the message packet is preferably analyzed at the lowest software level of the OSI stack, ensuring that the kernel examines the subelements of each message packet.

In one embodiment, each workstation 86 couples through a modem 64 to the Internet 55, and includes a kernel that performs security.

Figure 5 is a flow diagram of a method for establishing multilevel ports according to the present invention in which a requesting application runs on a first data processing node 45 (i.e., Machine One). A second data processing node 86 (i.e., Machine Two) includes a plurality of ports associated with predetermined security classifications. According to the present invention, Machine One runs 68 a selected application, which establishes its own security level consistent with the security clearance of the user. When the application being run calls a resource or object at another data processing node, the local machine kernel opens 70 a socket to the other resource or object for which a message carrying a service request can be made. The socket identifies the destination machine, a port number corresponding to the application program being run, and the local process security level. A port identifier is created by first requesting 72 an applicable security level for the associated port number opened by the kernel. The kernel further checks to see if the requested port is available 74 at that security level. If that port number and security level combination is currently in use (e.g., by another user) the kernel waits 76 for a predetermined time before again polling to determine if the particular security level is available for the port number. On the other hand, if the particular port number and security classification combination is available, the kernel combines the security level and port number to create 78 a port identifier. Then, the applicable IP header for a message packet is created 80 by inserting the port number and security label combination into the protocol spaces of the IP header normally reserved for just the port number. The message packet is completed by attaching 82 application specific data and information into predetermined regions of IP header to create a complete datagram. The completed datagram packet is then formatted 84 for electronic communication and sent to the destination server 86.

The operating system kernel 86 of data processing node 86 intercepts 88 the packet from Machine One and examines 90 the subelements of the packet to extract the port identifier. Once the port number and security label have been extracted, the kernel determines whether the requested port at the specified security level is in open status, and if so, whether it is presently available 92 for access. If the port is unavailable in that the combined port number and sensitivity label is in use by another application, then the operation terminates 93. If the port is available, applicable data from the message packet is transferred 94 to the applications portion of the applicable operating system stack of data processing node 86 for application processing. After data is provided to the application, an applicable reply is prepared 96 as appropriate, and an applicable IP header is attached 98 to the reply message which is prepared. The reply message is formatted 100 for packet transmission over an electronic network, and sent to first data processing node 45.

The kernel of first data processing node 45 intercepts 102 the applicable reply packet and examines the packet to verify 104 that the reply message has been provided at the same security level as the applicable application process is running in data processing node 44. If the security levels of the local process and the remote message received are the same, the reply is passed 106 to the application for processing. If the reply is at a security level inconsistent with the security level of an applicable local application, the reply packet is terminated and, if applicable, a negative acknowledgment is sent 108 to the second data processing node 86. Although the reply packet examination shown in Fig. 5

indicates that the security level of the reply packet is the same or equivalent to the security level of the application process, according to the present invention, the reply packet may have a lower security level if the reply packet is to be read by the application. Any access controls may be used for receipt of message packets so long as the control is consistent with the system's security policy.

Figure 6 shows a method according the present invention to determine whether a requested port is available for communication between data processing nodes. In particular, an incoming packet 86i is shown intercepted 110 by a destination system's operating system. Security examination is performed at the data link and network levels of the kernel interface operating system interface 66. The IP header element 112 of packet 86i' is examined and the port number and the security label subelement 114 are identified. The kernel checks to determine if the requested port number is already open 116. If not, the requested port is opened 118 at the security level indicated by the security label. Activities for opening a port at a particular security level are logged 122 to provide a journal or history of the activity and to provide a database of security levels which are presently open for particular port numbers. A decision is made 120 whether to pass the packet to a local application. If all other protocol requirements have been satisfied, the data is passed to the applications process 86" for handling and completion. If all other protocol requirements have not been satisfied, the packet is dropped 108.

If a registered port number requested is already open 116, the operating system kernel determines 124 whether each opened port is at the security level specified by the port identifier's security label. If not, then a new port having the same number as the existing port is opened 118 at the identified security level. The opening of the port is logged 122 to journal the activity, as described above. If the existing open port is at the same security level as identified in the port identifier subelement, then it is determined 126 whether the port is in use. If the port is presently in use, then a mandatory access control protocol precludes opening another port at the same number and security level being opened. Consequently, a packet is either buffered 128 and checked periodically until a pre-defined time-out 130 occurs, causing packet process termination or the packet is terminated 108 immediately, or until the port becomes unused 124, 125. If an open port is set to a correct security level but not currently in use 126, then the port activity is logged and a decision is made 120 whether or not to pass the packet. If all other security criteria is met, the packet is forwarded for application processing.

According to the present invention, a computer system having an operating system adhering to selected access control security mechanism includes government systems wherein a hierarchy of security classification levels are defined (e.g., top secret, secret, classified, unclassified), and commercial systems. For purposes of this application, sensitivity labels pursuant to an access control security mechanism includes at least hierarchical security classifications, as described above, and may include non-hierarchical categories or compartments. For example, these categories may refer to various plant sites according to particular demographics, product types, as well as categories defined by cross-functional boundaries such as accounting, public relations, marketing, engineering and R&D. Consequently, an entity holding a particular security classification may not automatically be cleared for all information at that level in every category. An application instantiated in the memory of the computer system may require access to a third party resource or object either on the same system or on a different system. The kernel, after determining that the user has permission to demand the resource, generates an IP header in preparation for communicating with the resource. The IP header includes source and destination machine identification numbers, and port identifiers. The port identifier for a destination system comprises a port number specifying a particular resource, database, or service requested by the source application, and a sensitivity label. The sensitivity label includes a security classification or clearance of the process in which the application is running, and may include other information such as category restrictions. The source system kernel attaches any application data to the header to create a datagram or message packet. The source system kernel further opens a communications socket and transmits the resultant packet to a selected destination system.

The destination system kernel receives the packet sent and analyzes the port identifier in the packet header. If the requested port number has not yet been opened on the destination system, the destination system kernel launches the requested application at a process security level consistent with the security level identified by the sensitivity label in the port identifier in the packet header (i.e., a same or lower classification level). The process run may further be qualified by a category designator carrying the security label of the source system packet, establishing multiple ports at the same port number and clearance for different categories. Packet examination and reading occurs according to one embodiment of the present invention at a destination system server, at a gateway server acting as a firewall between a destination server and a third party system, or at any server internetworked with the destination server.

Further according to the present invention, any requested jobs and services are performed. If the clearance of an object process is the same as the source process clearance, the destination system kernel creates a reply packet for transmission to the source computer system. However, if the destination system kernel

determines that the port number is open, but that the sensitivity label associated with the source is different from the sensitivity label of the opened port, the destination system kernel will open another port having the same port number at a security classification consistent with the sensitivity label of the source port identifier. Similarly, should another incoming packet have a source port identifier in its IP header request the opening of a third instantiation of the destination port at a third, different security classification, the destination systems kernel launches a third instantiation of the application pursuant to a process having a security classification consistent with the sensitivity label of the third port identifier. It is clear that as many instantiations of an application having the same port number may be opened, or running contemporaneously, as there are classification levels. Moreover, if additional categories are used to create unique port identifiers, then the number of ports having a common port number that might be opened contemporaneously is the sum of the number of categories.

If a destination system kernel determines that a port number is open at a particular classification level or for the same category and is open, the destination system kernel passes the received packet to an open destination process. However, if a destination port has the appropriate classification level or the same category is presently occupied with a previously received request, the destination system kernel does not pass the received packet to an associated destination process. Instead, the receiving kernel may buffer the received packet until a process becomes available at an acceptable security level or the kernel may reject the packet. An appropriate response message may then be sent back to the source system.

By way of example without limitation, an application instantiated in the operating system of a computer may require access to an external resource for license validation or verification. As a result, the receiving system operating system constructs a datagram or message packet comprising an IP header including source and destination socket identifications and communications protocols and may attach a license validation request associated with the application. A socket associated with a source process includes a machine address and a port number identifying a desired resource (e.g., the license validation service). According to the present invention, a new port identifier comprises a port number and a sensitivity label. Upon receipt of a message datagram or packet by a recipient license server, a receiving kernel examines the received message according to receiving system security protocols. The receiving kernel determines whether the port designated by the received message at the particular classification indicated by the sensitivity label in the message header is open. If the port at that classification is not open, or is unoccupied, then the kernel transfers the received message packet to a communications manager and opens a licensing verification application instantiated in a process at the indicated security label. If the port at the designated security classification has already been opened and is occupied (i.e., the required resource is in use by another user at the same security classification), the packet is buffered or dropped and a negative acknowledgment may be communicated back to the source system.

According to the present invention, security daemon resident in the receiving system executes a receiving system security protocol and determines whether to receive arriving message packets and whether to open a port at a requested security level. The security daemon according to one embodiment of the present invention operates between an Open Systems Interconnection (OSI) data link layer and OSI network. By inspecting incoming datagram and packet messages, the security daemon ensures that the kernel intercepts and inspects packets and messages traversing local interfaces. The security daemon according to the present invention accesses individual packet elements and sub-elements of the port identifier.

According to the present invention, multiple system sockets or endpoints having the same port number and a unique sensitivity label are opened to third party applications at network endpoints including multilevel trusted systems.

Although the invention is described herein in terms of preferred embodiments, it is understood that after having read the above description, various alternatives will become apparent to those persons skilled in the art. For example, the security label need not be associated with the port number at the source server. A composite port identifier according to the present invention, which comprises both port number and a security label, can be constructed at any time prior to the opening of a destruction port. Accordingly, software modifications at the source data processing need not include combining the security label with the port number. The port number may be associated with the data in a transmittal packet, and combined with the port number incident to examination by the destination server kernel. The present invention accordingly includes the scope of the appended claims stated as broadly as the prior art will permit and specification will permit.

Claims

1. A computer program product comprising:

   a computer useable medium having a computer readable program code mechanism embodied therein for generating a plurality of ports, said ports being associated with a common port number, each of said ports having a selected sensitivity label, said port number and said sensitivity label defining a selected port identifier for at least one of said ports, permit-

ting multiple, simultaneous access to the port, said computer code mechanism comprising:

first computer readable code mechanism for constructing a communications packet comprising a protocol header in turn comprising at least source machine identification, source port number, and destination port identifier region, said destination port identifier region including a destination port number and sensitivity label subregion; and

second computer readable code mechanism for permitting reception communications packets for establishing receiver ports.

2. A first program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to establish a multilevel port for enabling multiple, simultaneous access of a resource in a multilevel trusted system, said first program storage device comprising:

first computer readable code devices configured to receive a communications packet from a source machine running an application instantiated in a first process, said packet comprising at least a first destination port number and a first sensitivity label;

second computer readable code devices configured to examine said packet for identifying said port number and said sensitivity label, said port number and said sensitivity label, together providing a port identifier;

third computer readable code devices configured to compare said port identifier to port identifiers associated with pre-existing open ports; and

fourth computer readable code devices configured to open a port having the same port number as pre-existing open ports when said sensitivity label of said port identifier is unique as compared to sensitivity labels of pre-existing open ports, said opening permitting contemporaneous processes associated with a plurality of ports having the same port number, and a unique sensitivity label.

3. A first program storage device as in claim 2 further comprising a kernel having a security portion, said security portion including said third and fourth computer readable code devices.

4. A first program storage device as in claim 3, further comprising:

fifth computer readable code devices configured to pass a data portion of the communications packet to the process instantiating the application associated with the port previously opened in said port opening step;

sixth computer readable code devices configured to prepare a reply communication packet for transmission to said first process, said reply communication packet comprising at least a destination port number, a second sensitivity label, and a reply;

seventh computer readable code devices configured to transmit said reply communication packet to said source machine; and

eighth computer readable code devices configured to process said reply communication packet by said source machine in accordance with the security protocol of said source machine.

5. A computer having a multi-level trusted operating system, comprising:

a computer useable medium having a computer readable program code mechanism embodied therein for generating a plurality of ports, said ports being associated by a common port number, each of said ports having a unique sensitivity label, the combination of said port number and said sensitivity label defining a unique port identifier for each of said ports, said plurality of ports permitting multiple, simultaneous access of said common port number, said computer readable code mechanism in said multi level-trusted system.

6. A computer as in claim 5, wherein said computer readable code mechanism also includes computer readable code means for receiving a communications packet, for examining the packet to extract a destination port number and a sensitivity label, for determining the availability of a port having a unique port identifier address, and for opening a port having a unique port identifier address.

7. A multilevel port for permitting simultaneous access by a plurality of processes, each process having a different sensitivity label, the multilevel port defined by a common port number and a plurality of selected, unique sensitivity labels to permit two-way communication between said port and a plurality of processes having the same sensitivity labels.

8. A method for enabling simultaneous access of a port by a plurality of processes in a multilevel trusted system, comprising the steps of:

intercepting a first communications packet in a second computer system, said communications packet generated by the kernel of a first computer system, said communications packet comprising a destination port number and a

first sensitivity label;

examining the communications packet to extract and identify said port number and said sensitivity label, said port number and said sensitivity label combination defining a port identifier;

comparing said port identifier to the port numbers and sensitivity labels of pre-existing open ports;

establishing a port in the event no pre-existing open port has the same port identifier as defined in said communication packet;

passing the data portion of said communication package to an applications process in said second computer system, said applications process having a port number and sensitivity label equivalent to said port identifier.

9. A method for enabling simultaneous access of a port as in claim 8 further comprising:

preparing a reply;

constructing a second, return communications packet, said return communications packet comprising at least a reply, a source port number, and a second sensitivity label associated with said applications process in said second computer system;

transmitting said second communications packet to said first computer system;

intercepting said second communications packet by a kernel in said first computer system;

comparing said first sensitivity label to said second sensitivity label; and

processing the reply in accordance with the security protocol associated with the kernel in said first computer system.

10. A method for enabling simultaneous access of a port as in claim 8 wherein said intercepting step is performed by a daemon operating between the data link and the network layers of a second computer system operating under an OSI protocol.

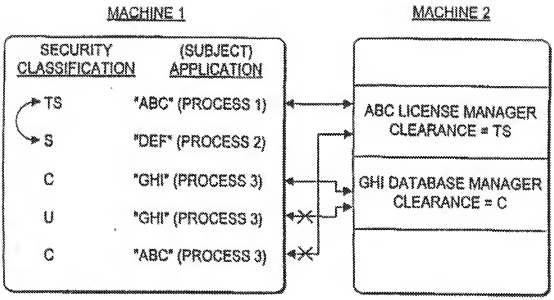


FIGURE 1
PRIOR ART

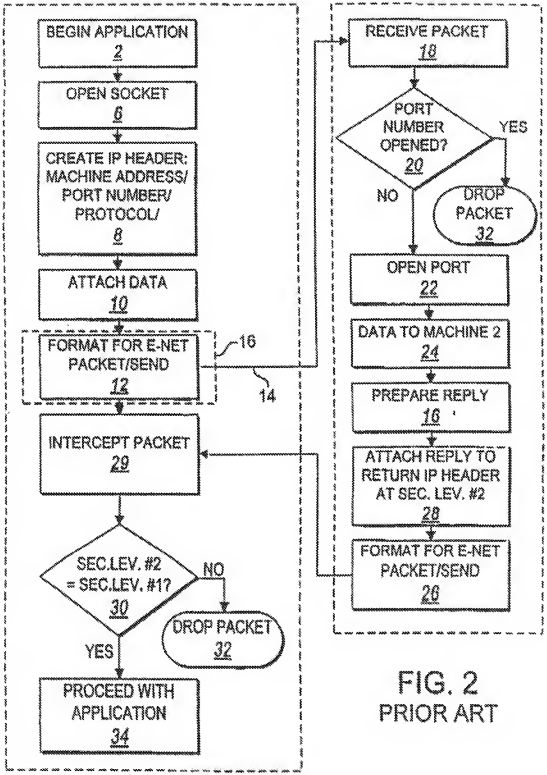


FIG. 2
PRIOR ART

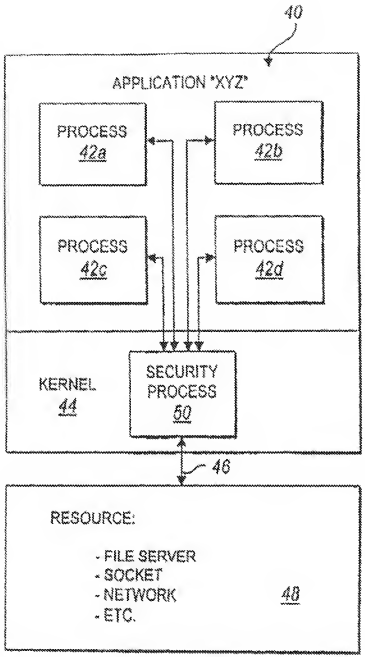


FIG. 3
PRIOR ART

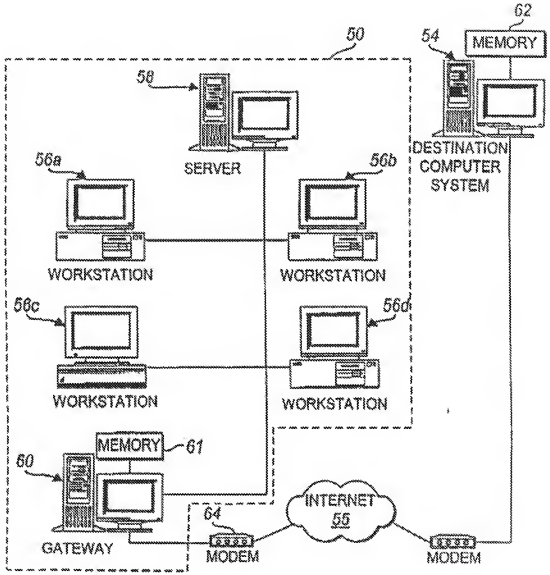

62
MEMORY
54
50
58
SERVER
DESTINATION
COMPUTER
SYSTEM
56a
56b
WORKSTATION
WORKSTATION
56c
56d
WORKSTATION
WORKSTATION
MEMORY
61
60
GATEWAY
64
MODEM
INTERNET
55
MODEM

FIG. 4

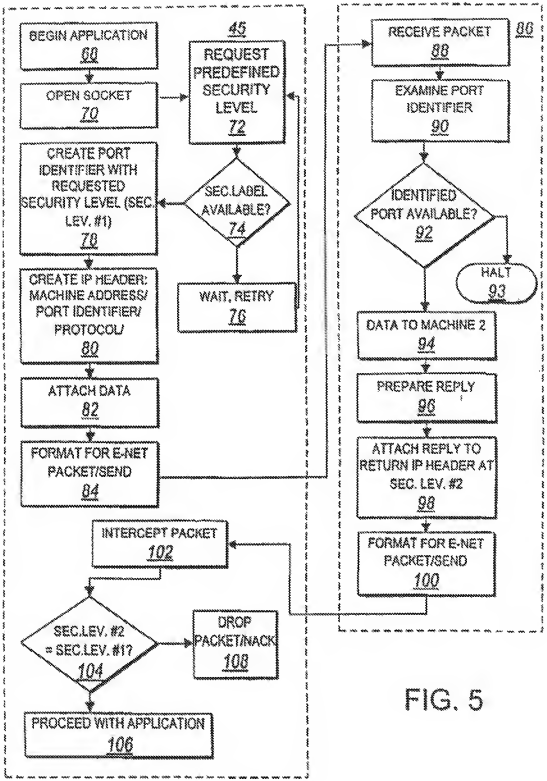

BEGIN APPLICATION
60
OPEN SOCKET
70
CREATE PORT IDENTIFIER WITH REQUESTED SECURITY LEVEL (SEC. LEV. #1)
78
CREATE IP HEADER: MACHINE ADDRESS/ PORT IDENTIFIER/ PROTOCOL/
80
ATTACH DATA
82
FORMAT FOR E-NET PACKET/SEND
84
45
REQUEST PREDEFINED SECURITY LEVEL
72
SEC.LABEL AVAILABLE?
74
WAIT, RETRY
76
INTERCEPT PACKET
102
SEC.LEV. #2 = SEC.LEV. #1?
104
DROP PACKET/NACK
108
PROCEED WITH APPLICATION
106
86
RECEIVE PACKET
88
EXAMINE PORT IDENTIFIER
90
IDENTIFIED PORT AVAILABLE?
92
HALT
93
DATA TO MACHINE 2
94
PREPARE REPLY
96
ATTACH REPLY TO RETURN IP HEADER AT SEC. LEV. #2
98
FORMAT FOR E-NET PACKET/SEND
100

FIG. 5

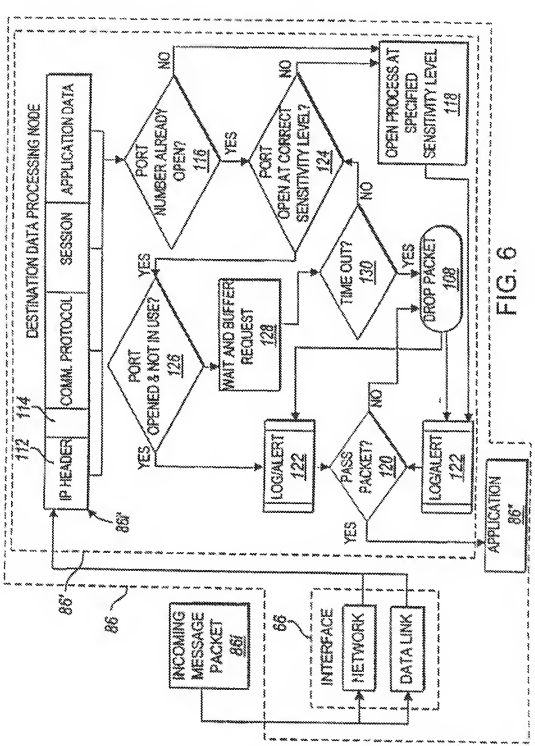

DESTINATION DATA PROCESSING NODE
112
114
IP HEADER
COMM. PROTOCOL
SESSION
APPLICATION DATA
86'
86
INCOMING MESSAGE PACKET 86i
66
INTERFACE
NETWORK
DATA LINK
PORT OPENED & NOT IN USE? 126
YES
NO
WAIT AND BUFFER REQUEST 128
LOG/ALERT 122
PASS PACKET? 120
YES
NO
LOG/ALERT 122
APPLICATION 86"
PORT NUMBER ALREADY OPEN? 116
YES
NO
PORT OPEN AT CORRECT SENSITIVITY LEVEL? 124
YES
NO
TIME OUT? 130
YES
NO
DROP PACKET 108
OPEN PROCESS AT SPECIFIED SENSITIVITY LEVEL 118

FIG. 6

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 893 751 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
27.01.1999 Bulletin 1999/04

(51) Int. Cl.[6]: G06F 1/00

(21) Application number: 97202854.2

(22) Date of filing: 16.09.1997

(84) Designated Contracting States
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV RO SI

(30) Priority 18.07.1997 EP 97202254

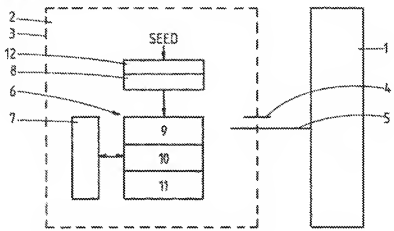
(71) Applicant: IRDETO B.V.
2132 HD Hoofddorp (NL)

(72) Inventor: Rix, Simon Paul Ashley
Randburg 2125 (ZA)

(74) Representative:
de Vries, Johannes Hendrik Fokke
De Vries & Metman,
Gebouw Autumn,
Overschiestraat 184 N
1062 XK Amsterdam (NL)

(54) **Integrated circuit and method for secure data processing by means of this integrated circuit**

(57) An integrated circuit comprises logic circuitry, such as a microprocessor, and a secure co-processor protected by a cocoon. The co-processor is connected to the logic circuitry at least by data lines. The co-processor comprises a cryptographic engine and a volatile storage element for storing a secret key.



EP 0 893 751 A1

Description

The present invention relates to an integrated circuit and to a method for secure data processing using a secret key by means of this integrated circuit.

In the areas of pay television, banking, or any other environment of secure data processing, the system or method used relies on the secure storage of a secret piece of information, i.e. a secret key. This secret key is used by a microprocessor for carrying out cryptographic functions. In the integrated circuits comprising a microprocessor and storage element for the secret key, security is obtained by making the entire microprocessor and storage element secure by encapsulating the complete circuitry in a cocoon, labyrinth or encapsulation which may comprise power, ground and/or sense wires. However encapsulating the complete circuitry is rather complex in view of the area size of the circuitry in the integrated circuit chip. Further, there are several connections between the microprocessor and peripheral circuitry, each of these connections being a possible point of attack for unauthorized persons to obtain information which can be used in finding the secret key. Moreover, if the operation of the microprocessor is stopped during a cryptographic operation, the registers of the microprocessor contain information which can be used by unauthorized persons to derive the secret key. It will be clear that as soon as the secret key has been found by unauthorized persons, the security of the system has been broken.

The invention aims to provide an integrated circuit and method of the above-mentioned type with enhanced security.

According to the invention an integrated circuit is provided, comprising logic circuitry, preferably a microprocessor, and a secure co-processor protected by a cocoon, said co-processor being connected to said logic circuitry at least by data lines, wherein said co-processor comprises a cryptographic engine and a volatile storage element for storing a secret key.

In this manner an integrated circuit is provided wherein only a small part of the circuitry, i.e. the secure co-processor or secure cell, needs to be encapsulated in a cocoon, which in view of the small size of the co-processor area is possible in a relatively easy manner with high security. All storage and cryptographic functions are contained within the cocoon, so that no part of any cryptographic process is visible to any external means at any stage of its operation. Only messages from the microprocessor to be processed by the secure co-processor and processed messages are available on the data lines, which information however provides no information at all regarding the secret key. In this manner it is impossible for any third party to find any information on the secret key in an attempt to break the security.

As the secret key is stored in a volatile storage element, any attempt to access the secure co-processor will result in a loss of the secret key as such an attempt will be detected by the cocoon resulting in a loss of power and thereby in erasure of the secret key.

According to the invention a method for secure data-aprocessing is provided using a secret key, comprising the steps of loading the secret key in the storage element, sending encrypted information from the microprocessor to the co-processor via the data lines together with control information, using the secret key to decrypt said information in accordance with the control information in the co-processor, authenticating the decrypted information, and using the decrypted information in accordance with the control information.

The invention will be further explained by reference to the drawing in which an embodiment of the integrated circuit according to the invention is shown in a very schematical manner.

By way of example it will be assumed that the integrated circuit shown is part of a smart card used in the conditional access module of a decoder system for pay television. However, the invention is certainly not restricted to such an application. On the contrary, the invention can be used in a wide area of cryptographic applications.

The integrated circuit comprises a microprocessor 1 and a secure co-processor 2 encapsulated in a cocoon 3 of security wires which may include power, ground and/or sense wires. The security wires 3 are indicated by a dashed line surrounding the co-processor 2. In the actual integrated circuit the co-processor 2 will be covered substantially completely by the security wires at least at the top and bottom sides. It is observed that the term cocoon as used in this specification can be a labyrinth, cover or encapsulation of power, ground and/or sense wires or another active or passive means preventing access to the co-processor 2.

The co-processor 2 is connected to other circuitry of the integrated circuit, in particular to a clock circuit not shown and to the microprocessor 1 by clock and data lines 4, 5. The co-processor 2 comprises a cryptographic unit 6, a control unit 7 and a volatile storage element 8 for storing a secret key. The cryptographic unit 6 comprises a decryption engine 9, an authentication engine 10 and preferably also an encryption engine 11. Further, the cryptographic unit 6 includes a one-way function block 12 to load the secret key into the storage element 8. Power consumption of the elements of the secure co-processor 2 is very low and power is provided by a battery not shown.

The storage element 8 for the secret key and all cryptographic functions are contained within the cocoon 3, so that no part of any cryptographic process is accessible to any external means at any stage of operation of the coprocessor 2. The actual decryption, encryption and/or authentication functions are no part of the present invention and therefore a detailed description of such functions is not necessary. Any decryption, encryption or authentication normally used in crypto-

graphic processes can be implemented in the logic circuitry of the co-processor 2. It is noted, however, that the number of logic elements used for the co-processor 2 is preferably as small as possible as this will result in a small cocoon with very high security.

Any attempt to enter the cocoon 3 will result in a contact with any of the sense wires or a short circuiting of ground and power wires so that the power of the co-processor 2 will be disconnected. Such an attempt would therefore lead to an erasure of the secret key stored in the storage element 8.

Using the integrated circuit described, data-processing is possible in a very secure manner by first loading a secret key in the storage element 8 by sending a seed through the one-way function block 12 to the storage element. As the secret key is loaded through the datalines 5 to the secure co-processor using the one-way function, for example a one-way hash function, the smart card cannot be re-used even if a secret key has been determined by unauthorized persons, as the one-way function is unknown.

After loading the secret key in the storage element 8, the microprocessor 1 can request the co-processor 2 to decrypt encrypted information forwarded via the datalines 5 to the co-processor 2 together with control information to indicate the requested operation to the control unit 7, and an authentication vector. The co-processor 2 uses the secret key to decrypt the information and the decrypted information is authenticated in a usual manner. The decrypted information is thereafter used by the co-processor 2 in accordance with the control information and this control information can either indicate that the decrypted information should be returned to the microprocessor 1 or should for example be used as a key for a next decryption step on a next encrypted information message from the microprocesor 1. In this latter case a chain of two or more decryption steps can be performed within the co-processor 2 without returning decrypted information to the microprocessor 1.

The control information in the messages provided by the microprocessor can contain information as to which decryption or encryption algorithm is to be used by the co-processor 2 and any other required configuration information.

It is noted that although the co-processor 2 is shown as comprising a number of separate blocks the actual implementation of this co-procesor can be made in any suitable manner.

The invention is not restricted to the above-described embodiment which can be varied in a number of ways within the scope of the claims.

Claims

1. Integrated circuit, comprising logic circuitry, preferably a microprocessor, and a secure co-processor protected by a cocoon, said co-processor being connected to said logic circuitry at least by data lines, wherein said co-processor comprises a cryptographic engine and a volatile storage element for storing a secret key.

2. Integrated circuit according to claim 1, wherein said secure co-processor comprises a one-way function unit, wherein a secret key is loaded in said storage element by providing a seed to said one-way function unit.

3. Integrated circuit according to claim 1 or 2, wherein the cryptographic engine comprises a control unit, a decryption engine and an authentication engine.

4. Integrated circuit according to claim 3, wherein the cryptographic engine further comprises an encryption engine.

5. Integrated circuit according to anyone of the preceding claims, wherein said cocoon comprises security wires, preferably including power, ground and/or sense wires.

6. Integrated circuit according to anyone of the preceding claims, wherein at least the volatile storage element for the secret key is powered by a battery.

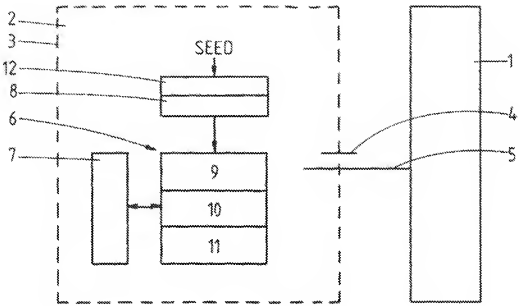
7. Method for secure data processing using a secret key by using an integrated circuit according to anyone of the preceding claims, comprising the steps of

   - loading a secret key in the storage element,
   - sending encrypted information from the microprocessor to the co-processor via the data lines together with control information,
   - using the secret key to decrypt said information in accordance with the control information in the co-processor,
   - authenticating the decrypted information, and
   - using the decrypted information in accordance with the control information.

8. Method according to claim 7, wherein the decrypted information is used as decryption key in the co-processor to decrypt further encrypted information received from the microprocessor.

9. Method according to claim 7 or 8, wherein the decrypted information is returned to the microprocessor.

10. Method according to claim 7, 8 or 9, wherein the secret key is loaded into a storage element by applying a one-way function on a seed.


2
3
12
8
6
7
SEED
9
10
11
1
4
5

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 97 20 2854

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	WO 96 00953 A (NAT SEMICONDUCTOR CORP) * page 1, line 6 - page 6, line 8 * * page 10, line 25 - page 11, line 12 * * page 12, line 26 - page 14, line 11 * * page 30, line 24 - page 33, line 7 * * page 49, paragraph 4 * * page 53, line 6 - page 55, line 9; figure 1 *	1-10	G06F1/00
X	US 5 515 540 A (GRIDER STEPHEN N ET AL) * column 1, line 62 - column 2, line 8 * * column 23, line 17 - line 63 * * column 28, line 49 - column 30, line 41; figures 1,11-14 *	1,2,5-10	
X	MORI R ET AL: "SUPERDISTRIBUTION: THE CONCEPT AND THE ARCHITECTURE" TRANSACTIONS OF THE INSTITUTE OF ELECTRONICS, INFORMATION AND COMMUNICATION ENGINEERS OF JAPAN, vol. E73, no. 7, July 1990, pages 1133-1146, XP002010383	1,5,6	
A	* page 1138, left-hand column, line 1 - right-hand column, paragraph 4 * * page 1143, left-hand column, line 1 - right-hand column, paragraph 2; figures 4,8 *	2-4,7-10	TECHNICAL FIELDS SEARCHED (Int.Cl.6) G06F
A	EP 0 750 410 A (NIPPON TELEGRAPH & TELEPHONE) * page 6, line 10 - page 7, line 38; figures 3,4 *	1-10	

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	17 December 1997	Moens, R

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 926 605 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
30.06.1999 Bulletin 1999/26

(51) Int. Cl.[6]: G06F 17/30

(21) Application number: 97309328.9

(22) Date of filing: 19.11.1997

(84) Designated Contracting States
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant:
Hewlett-Packard Company
Palo Alto, California 94304 (US)

(72) Inventors:
- Edwards, Nigel
Horfield, Bristol BS7 8SR (GB)
- Rees, Owen
Bassaleg, Newport NP1 9LU (GB)
- Zhong, Qun
Bradley Stoke, Bristol BS32 8AY (GB)

(74) Representative:
Lawman, Matthew John Mitchell et al
Hewlett-Packard Limited,
IP Section,
Building 2,
Filton Road
Stoke Gifford, Bristol BS12 6QZ (GB)

(54) Browser system

(57) A Web browser (210) is configured to run in a middle compartment (206) of a compartmented mode workstation (CMW) (200). The operation of the Web browser (210) is prevented from accessing or damaging other compartments of the CMW machine (200) as a result of mandatory access control (MAC), which is configured appropriately.

The Web browser (210) communicates with Web servers (252) attached to the Internet (240), the Internet being connected to an outside compartment of the CMW machine (210), via a trusted outside process (TPO) (214). TPO (214) has the privileges required to override MAC. The Web browser (210) communicates with a display server (232), which is attached to an inside compartment (204) of the CMW machine (210), via a trusted inside process (TPI) (204). TPI also has privileges to override MAC. The Web browser (210) can request and receive Web pages incorporating mobile code, and can process the mobile code safely within the middle compartment (206). As a result of processing the mobile code, the Web browser (210) sends only X-messages to the display server (232), in order that the display server can render the images resulting from the processed mobile code.

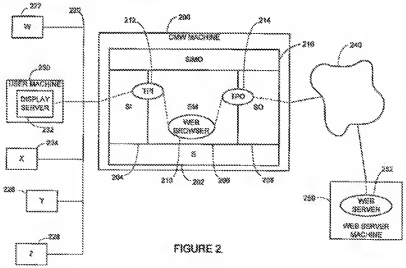


FIGURE 2.

Description

Technical Field

[0001] The present invention relates in general to computerised systems for down-loading, or 'browsing', information stored in computer-readable form. More particularly, although not exclusively, the invention relates to a browser system for browsing information that contains mobile code retrievable from the World Wide Web.

Background Art

[0002] The World Wide Web (Web) may be thought of as a global village where computers (hosts) are the buildings, and the worldwide computer network known as the Internet forms the streets. The computers have addresses (IP Addresses) consisting of four numbers separated by periods. Many hosts also have nicknames known as domain names. A Web site typically consists of a UNIX or Microsoft Windows based Web server, which runs on a host and 'serves' software or content to other computers accessing the Web site. A Web site is not a single application, but a system that provides access to applications and data stored on the host, as well as inside an organisation. A user utilises a Web 'browser' running on a client computer to access the software or content on the Web server.

[0003] Figure 1 illustrates a client computer 100 executing a Web browser program 105 that is employed by a user to communicate over the Internet 110, in a special language called HyperText Transfer Protocol (HTTP) 115, with a host computer 120 executing a Web server program 125 to obtain data. Hereafter, the term 'Web browser' may be used interchangeably to describe a Web browser program or the program in execution on a computer, depending on the context. In the diagram, and in following diagrams, solid connection lines represent physical connections between hardware and broken connection lines represent logical connections between software processes. The most basic Web transaction involves the transmission of Web pages, written in HyperText Markup Language (HTML) from the Web server 125 to the Web browser 105. Upon request by the user at the Web browser 105, the Web server 125 translates the HTML-based Web page into HTTP and sends it over the Internet 110 for display as a Web page on the requesting browser 105. The Web browser 105 receives the HTTP-encoded Web page, translates the HTTP back into HTML and displays the page.

[0004] The concept of 'mobile code' has been developed to extend the functionality of the Web. Mobile code is typically code associated with a Web page which, when downloaded from a Web server, automatically executes within the environment of the requesting Web browser. In a simple form, mobile code can be used to enhance the graphical appearance of a Web page by, for example, implementing simple animation. It is envisaged, however, that mobile code will be used to implement many different and far more complex functions in future. A good example of one use for mobile code is to download transactional clients, which support specialised user interfaces, to support data transfer between client and server applications.

[0005] Commonly, mobile code is written in the Java programming language as a Java applet. Mobile code may also be written in other languages, such as defined in the ActiveX model. Both Java applets and ActiveX control functions can be embedded into a standard Web page. Therefore, the simple operation of downloading a Web page can also download and activate associated mobile code.

[0006] While mobile code can greatly extend the functionality of the Web, the same extended functionality, by its nature, leads to serious security issues.

[0007] Mobile code, and Web browsers that run mobile code, are developed according to rigid security guidelines which are intended to prevent the possibility that malicious users can use mobile code to cause harm to the computing environment surrounding a Web browser. However, there are already many documented flaws in the security measures, which can lead to devastating results. Typically, the party downloading 'rogue' mobile code would be unaware of the damaging effect thereof until it was too late.

[0008] Some serious mobile code attacks known take advantage of bugs in the mobile code processing environment of the Web browser, which allow the mobile code to gain control over the operating system of the computing platform. From this position, the mobile code could cause damage such as deleting all files on the computer, or even launching attacks on other, networked computing platforms.

[0009] Other serious mobile code attacks are known as 'social engineering' attacks. These attacks rely on tricking an unwary user by, for example, sending the user a 'patch' for the Web browser, and suggesting that the patch is to remedy a security flaw in the Web browser. The patch, instead of being one that remedies a security flaw, actually overwrites good code with code that creates a security flaw. There are many other ways of tricking unwary users in this way.

[0010] Web browsers, which can run mobile code, such as Netscape Navigator™, typically include the option to 'disable' mobile code processing, thereby preventing the potential for any damage, even if mobile code is downloaded. Of course, this radical measure, whilst being very effective, also removes any benefit which can be obtained from genuine, safe mobile code.

[0011] It would therefore be desirable to have a system in which mobile code can be executed safely, while at the same time not allowing rogue mobile code to cause any damage to any system.

Disclosure of the Invention

[0012] In accordance with a first aspect, the present invention provides a secure browser system as claimed in claim 1.

[0013] The term browser is commonly associated with complex and sophisticated programs such as Netscape Navigator™ or Internet Explorer™. These programs are well known. However, herein, the term browser is used more broadly to include any program or system which, when running, is able to receive a requested resource, for example a Web page, from a source such as a Web server connected via a communications network to the browser. Further, a browser according to the present invention can even receive unsolicited resources as a result of, for example, some form of 'push' technology, which distributes resources or messages to registered subscribers.

[0014] The invention has the advantage that mobile code is processed in a secure environment, so that the client, which is apart from the environment, remains relatively safe from attack. The client only receives data from the browser to visualise the output of the processing of the mobile code on the browser. The client is, therefore, in effect able to access mobile code, and see the result of the processing of the mobile code, without being subjected to any threat from rogue mobile code.

[0015] In a preferred embodiment of the present invention, the browser system comprises a secure operating system, for example one which enforces Mandatory Access Control (MAC), such that mobile code and the browser are unable to damage the system running the browser, let alone the client.

[0016] While the invention, in general, aims to protect user systems from rogue mobile code, and from vulnerable browsers running rogue mobile code, embodiments which employ secure operating systems, such as those providing MAC, can be configured to also provide a high level of protection to the computer platform that supports the browser running the mobile code. Such systems consequently can provide even more protection to users' systems, by greatly reducing the risk of mobile code reaching users' systems, or other parts of the network, by some other route.

[0017] Other aspects and features of the invention are described and claimed below.

Brief Description of the Drawings

[0018] A preferred embodiment of the present invention will now be described, by way of example only, with reference to the accompanying drawings, of which:

Figure 1 is a diagram illustrating a standard Web environment;

Figure 2 is a diagram illustrating a CMW machine configured for operation in accordance with the present embodiment;

Figure 3 is a diagram, which illustrates the 'dominates' relationships between compartments defined in the CMW machine of Figure 2;

Figure 4 is a diagram, which illustrates the relationships, and protocols that exist between the processes that operate for the purposes of the present embodiment;

Figure 5 is a flow diagram which illustrates the steps required to initiate a CMW machine for operation in accordance with the present embodiment; and

Figure 6 is a flow diagram, which illustrates the steps involved for the purposes of the present embodiment when a client requests a Web page including mobile code.

Best Mode For Carrying Out the Invention, & Industrial Applicability

[0019] According to the present embodiment, the Web browser operates on a computing platform within the environment of a secure operating system that enforces MAC. A particularly suitable secure operating system is the HP-UX 10.09.01 Compartmented Mode Workstation (CMW) sold by Hewlett-Packard Company, which provides a MAC policy governing the way data may be accessed on a trusted system.

[0020] The MAC policy is a computerised version of the US Department of Defence's long-standing multi-level security policy for handling classified information. The MAC policy uses labels that reflect information sensitivity, and maintains those labels for every process and file system object to prevent users not cleared for certain levels of classified information from accessing it. Under MAC, users and processes are also assigned clearances. A clearance defines the maximum sensitivity label the user or process can access, which is necessary since some users and processes have privileges that allow them to switch between sensitivity labels. Using the MAC policy, the operating system controls access based on the relative sensitivity of the applications running and the files they access. The HP-UX CMW operating system rates as a B1 grade secure operating system, according to the Orange Book [NCSC] criteria. In general B1 and higher-grade operating systems apply some form of MAC.

[0021] The HP-UX 10.09.01 CMW [DIA 91], is described in detail in the documents referenced at the end of this description, which are available from Hewlett-Packard Company. At the time of writing this description, HP-UX 10.09.01 CMW is the current version of the operating system. Future versions of the operating system, and the respective documentation, will, however, remain relevant to the present description and embodiment.

[0022] Hereinafter, for convenience of description only, the term "CMW machine" is intended to mean a computing platform with an operating system having

additional, CMW security features, which are described below. A particularly suitable operating system is Hewlett-Packard Company's HP-UX CMW operating system.

[0023] The following description describes in detail how to use Mandatory and Discretionary Access Controls, Sensitivity Labels, Trusted Processes and Privileges on a CMW machine to restrict the behaviour of mobile code and of a Web browser that downloads this code. A preferred arrangement is shown in Figure 2.

[0024] Figure 2 illustrates a CMW machine 200 connected via an internal network 220 to a user machine 230 running a display server 232, and via an external network to a Web server 252 machine 250 running a Web server 252. The internal network 220 is also shown connected to other apparatus, labelled w, x, y and z (labelled 222, 224, 226 and 228 respectively), which can be other user machines, servers or network appliances such as printers. The external network comprises a connection from the CMW machine 200 to the Internet 240 (via appropriate switching and routing equipment, which is not shown). The user machine 230 can be, for example, a PC, a UNIX workstation or an X terminal. For the present purposes, the user machine 230, in whatever form, is running an X display server 232. The internal network 220 comprises an Ethernet, which supports TCP/IP communications between the user machine 230 and the CMW machine 200.

[0025] The CMW machine 200 is configured to have one classification: System (S) 202; and three compartments: Inside (I) 204, Middle (M) 206 and Outside (O) 208. This generates eight sensitivity labels (the operation of which will be described in detail below) of which only five are used in Figure 2: S, SI, SM, SO, SIMO (shown as 216). The three other possible sensitivity labels - SIO, SIM, and SMO - are unused in this embodiment. The CMW machine 200 incorporates a Web browser 210, which is arranged to run in the SM compartment. The Web browser 210 in this case is a Netscape Navigator™ browser. A compartment is, in effect, a virtual machine within which processes and file objects associated with the virtual machine can operate or be operated on.

[0026] The display server 232 is attached to the SI compartment of the CMW machine 200, and the external network is attached to the SO compartment of the CMW machine 200. Thus, data received from, or transmitted onto, the external network acquires the sensitivity label of the SO compartment. Also, data sent to or received from the display server 232 acquires the sensitivity label of the SI compartment.

[0027] As already mentioned, sensitivity labels are associated with every process and file system object, and are used as the primary basis for all MAC policy decisions. A sensitivity label represents the sensitivity of a process or a file system object and also the data each contains. If an application and the file it attempts to access have compatible sensitivity labels, the application can read, write, or possibly execute the file, and each new process typically inherits the sensitivity label of its parent. For example, if a program is executed within a shell (for example, sh(1), csh(1), or ksh(1)), the new process automatically inherits the sensitivity label of the shell process. New files always inherit the sensitivity label of the process that creates them. The system can provide special trusted programs that may be employed for changing the sensitivity label of a file after it has been created.

[0028] Sensitivity labels are prioritised for MAC in a way that determines how processes or objects having one sensitivity label can interact with processes or objects having different sensitivity labels. The prioritisation is defined internally of the operating system. The diagram in Figure 3 represents the relationship between the parts of the system illustrated in Figure 2.

[0029] In Figure 3, the arrows point from dominating sensitivity labels to dominated sensitivity labels. Thus, in Figure 3: SIMO dominates SI, SM and SO; SO dominates S; SM dominates S; and SI dominates S. It should be noted that SO, SM and SI have no 'dominates' relationships between them. Also, the labels SMO, SIO and SIM, which are not used in the present embodiment, are illustrated for completeness in boxes with dashed lines to indicate where they would appear. One further important aspect of the dominates relationships, which is not shown in the diagram, is that each sensitivity label dominates itself.

[0030] Users are generally not permitted to downgrade (by reducing the respective sensitivity labels of) any files, processes or objects which they control, so that the new label is dominated by the previous label. Also, users are not permitted to cross grade them so that the new label is incomparable to the previous one. The system is also configured so that downgrading and cross grading are not enacted automatically by the acts of reading or writing.

[0031] The effect of the MAC policy is to rigidly control information flow in the system, from process to file to process, to prevent accidental or intentional mislabelling of sensitive information. To achieve this, for every operation, the system compares sensitivity labels to determine if a user or process can access an object. Any time a user or process tries to read, write, or execute a file, the system examines the process and object sensitivity labels and consults its MAC rules. For each operation a process requests the system determines if the process has mandatory read or mandatory write access to the object. Most restrictions that the MAC policy enforces can be summarised by the two following rules:

(1) Mandatory read access: a process can read or execute a file, search a directory, or (subject to other privilege requirements) read the contents of other objects if the process's sensitivity label dominates the object's. All of these operations involve transferring data from the object to the process, so

having such access is referred to as "mandatory read" access.

(2) Mandatory write access: a process can write to a file, remove or create an entry in a directory, or change any object's security attributes (including its sensitivity label), if the process's sensitivity label is the same as the object's. All of these actions involve transferring data from the process to the object, so having such access is called "mandatory write" access.

[0032] The first rule prevents a user who is not cleared for classified information from seeing it. The second rule prevents a user with a high clearance from revealing information to other users with lower clearances.

[0033] In effect, MAC in the CMW machine 200 ensures that information can flow only in the opposite direction to the "dominates" relationship. Thus MAC allows the mobile code and Web browser 210 to read data only with a sensitivity label of "S" or "SM". The Web browser 210 and mobile code can write data only with a sensitivity label of "SM". Neither the Web browser 210, nor the mobile code, is able to gain direct access to either the inside network or the outside network, since these have sensitivity labels of "SI" and "SO".

[0034] The CMW machine 200 does not impose the concept of an all-powerful "Super User" (e.g. "root") or Administrator. Instead, this power is divided up into a number of privileges. Assigning privileges to a program confers on it power to do particular actions. Programs with these privileges are known as 'trusted processes'. Trusted processes, TPI (trusted process - inside) 212 and TPO (trusted process - outside) 214, shown in Figure 2, have the privileges that allow them to override the MAC. Thus the Web browser 210 and mobile code must use TPI 212 and TPO 214 to gain access to the internal and external networks.

[0035] Trusted processes are typically very small programs, which are carefully designed to carry out a single, specific process, such as passing specific data between compartments in a CMW machine. Trusted processes have privileges which enable them to override MAC, but these privileges are only raised when required, and lowered thereafter, to minimise the chances of misuse by any other user or process. Also, a trusted process checks whether a user or other process has the right to access it, before allowing such access.

[0036] TPI 212 is a trusted process that manages the interaction between the real Web browser 210 (and mobile code) running in the SM compartment, and the display server 232 running on the inside network. In some embodiments, the display server 232 could in fact be running in the SI compartment on the CMW machine 200, but this would be less likely in a networked environment. TPI 212 has the necessary privileges that enable it to override MAC and pass data between the SI and SM compartments.

[0037] TPO 214 is a trusted process, which manages interaction between the real Web browser 210 (and mobile code) running in the SM compartment, and the Internet 240, which is connected to the SO compartment.

[0038] All messages from the Web browser 210 (and mobile code) to the external network are sent via TPO 214. TPO 214 can be configured to block undesirable messages from the Web browser 210, such as attempts to communicate with prohibited external sites or attempts to download mobile code from certain sites. Additionally, TPO 214 can be configured to block messages emanating from the downloaded code when it executes in the Web browser 210. TPO 214 can also be configured to filter incoming messages intended for the Web browser 210, in a similar fashion to a packet filter or firewall. The Web browser 210 runs in the SM compartment without any privileges. The Web browser 210 is configured to direct every network connection to TPO 214 by making use of built-in SOCKS functionality. That is to say, the Web browser 210 must support SOCKS, as will be described below.

[0039] The Web browser 210's executable file, the files, directories and the resources that are only read by the Web browser 210, such as the configuration files, are given the label S. The result is that the MAC protects these resources so that users, a broken browser or malicious mobile code cannot bypass the security administration by overwriting them with their own copies of these files. Other files that need to be both read and written to by the Web browser 210, such as a bookmark file, history files or a cache, are labelled as SM.

[0040] All the users and hosts of the internal network 220 are given the label SI and all the hosts of the Internet 240 are given the label SO. Since the Web browser 210 has no privileges, it and all its child processes, such as those executing mobile code, can only run with the label SM. Therefore, the behaviour of the Web browser 210 and mobile code is encapsulated in the SM compartment.

[0041] Thus, the CMW machine 200 configuration shown in Figure 2 ensures that the Web browser 210 running in the SM compartment cannot interfere with other processes running with other sensitivity labels. This configuration can be generalised to an arbitrary number of middle compartments (Middle_1,... , Middle_n). Each compartment can be used to isolate a Web browser 210 and any associated mobile code accessed by a user connected to the CMW machine 200. If some controlled sharing of information between the code in the Web browsers is required, multiple browsers can run in the same compartments, under different user identifiers. The CMW machine 200 therefore acts as a Web browser 210 server to let multiple users on the inside network use Web browsers 210 and mobile code securely and conveniently. Each user has a personal copy of his or her Web browser 210 resources, such as a bookmark file, on the CMW machine 200, with these resources all having the same sensitivity label.

Conventional Discretionary Access Control (DAC), as found in general operating systems such as UNIX, can be used to specify which local files owned by one user the mobile code downloaded by another user can access.

[0042] The implementation of the above architecture consists of the four components, three of which have been described above, namely: TPI 212, TPO 214 and the Web browser 210. These three components, and a fourth component, the trusted browser front end (TBFE), are illustrated in Figure 4. Figure 4 shows the relationship between these four components and the communication protocols in use between them.

[0043] The diagram in Figure 4 shows that the TBFE is a parent process to TPI 212, TPO 214 and the Web browser 210. In other words, TPI 212, TPO 214 and the Web browser 210 are child process to the TBFE. Communications between TPI 212 and the Web browser 210 comprise X-messages, communications between the Web browser 210 and TPO 214 comprise SOCKS messages and the communications between TPI 212 and the display server 232 comprise X-messages.

[0044] The following six privileges are defined within CMW and are used in accordance with the present embodiment to support the present system:

Allowmacread: overrides MAC restrictions on read operations, allowing a process having this privilege to read an object's data and attributes regardless of the object's sensitivity label;

Allowmacwrite: overrides MAC restrictions on write operations, allowing a process having this privilege to write an object's data and attributes regardless of the object's sensitivity label;

Chsubjsl: (stands for change subject sensitivity label) allows a process having this privilege to change its own sensitivity label to any label dominated by the process's clearance;

Configaudit: required by the ioctl(2) interface and used to configure the security audit system;

Suspendaudit: if raised, the security audit system does not produce system call records on behalf of the processes. Most trusted processes raise this privilege because they produce their own audit records, making those automatically generated by system calls unnecessary; and

Writeaudit: Required by the write(2) interface of the audit device to append records to the audit trail.

[0045] TPI 212 comprises a proxy display server (in this embodiment, a proxy X-server [X Window]). The Web browser 210 in effect sends all X requests needed to render itself on a screen to TPI 212, rather than to a local display server. Subsequently, TPI 212 forwards the requests to the remote display server 232 running on the user machine 230. TPI 212 can also be configured to filter out undesirable or dangerous messages before forwarding them to the remote display server 232 on the internal network 220. For example, TPI 212 may be configured to connect only to a predefined set of hosts or clients, and only to the display servers on those hosts. The details of such a configuration are beyond the scope of the present description, but are within the limits of ability of the skilled person.

[0046] For operation in accordance with the present embodiment, TPI 212 requires the Chsubjsl privilege to allow it to receive connections from both the SI and SM compartments. TPI 212 also requires the Allowmacread and Allowmacwrite privileges, so that it can pass data between the SM and SI compartments. TPI 212 also needs Configaudit, Suspendaudit and Writeaudit privileges to configure, manipulate and write audit records, as mentioned above.

[0047] TPO 214 comprises a connection request proxy, which in the present embodiment is a modified SOCKS server that uses the SOCKS [SOCKS] protocol to communicate with the Web browser 210, and mobile code downloaded by the Web browser 210, in the SM compartment. SOCKS is a well known, freeware proxy server, used to relay TCP streams between a client and the Internet 240. It is known to configure and use SOCKS as a filter or firewall application.

[0048] SOCKS is modified in TPO 214 in the present embodiment so that it can accept connections originating from multiple sensitivity labels. That is, TPO 214 can accept connections from the SM compartment, as well as from the SO compartment. This is achieved, as with TPI 212, using the Chsubjsl privilege. TPO 214 can also pass the data between compartments having different sensitivity labels using the Allowmacread and Allowmacwrite privileges subject to the security criteria set up by the system's security administrator.

[0049] TPO 214 also needs Configaudit, Suspendaudit, and Writeaudit to configure, manipulate and write audit records.

[0050] The process for initialising a Web browser 210 and its associated proxies will now be described with reference to the flow diagram in Figure 5.

[0051] In step 500, a TBFE is started remotely by the user, who has an account on the CMW machine 200, which authorises the user to activate a Web browser 210. The user can start TBFE by making use of remote execution functions provided by UNIX, such as 'remsh' or 'rexec'. To do this, the user would first have to be logged-on to the CMW machine 200. The server version of these functions can be rewritten to take the advantage of the CMW machine 200 to enhance security, but a description of how to achieve this is outside the scope of this text. A shell script to start the TBFE on the CMW machine 200 is installed on the user's machine. An alternative to a shell script would be to use Secure Shell

(SSH) to provide a secure login. Conveniently, SSH also encrypts the X-protocol messages, by using the SSH server (on the CMW machine 200) to pass the X-messages to the SSH client (on the user's machine). The SSH client then forwards the X-messages to the X-server running on the user's machine.

[0052] In step 510, when TBFE is started, it reads and parses its configuration file to check for semantic errors. An exemplary configuration file is reproduced below:

```
# lines start with # are comments
#
# start tpi at system inside and let it talk with system
middle
BEGIN_INIT{
#location of the program
PROGRAM: /home/proj 1/tpi
#sensitivity label to start the program
LEVEL: SYSTEM INSIDE
#arguments passed to the program
ARG: -1 "SYSTEM MIDDLE" -s zhong-q- 1 -n 1
}END_INIT
#
#start tpo at system outside
BEGIN_INIT{
PROGRAM: /home/proj 1/tpo
LEVEL: SYSTEM OUTSIDE
#arguments passed to program to define SOCKS
options
ARG: -d 3 -s
}END_INIT
#
#start netscape at system middle with the tpi as the
x-server
BEGIN_INIT{
PROGRAM: netscape
LEVEL: SYSTEM MIDDLE
#Netscape configuration argument
ARG: -display localhost: 1
}END_INIT
#
```

[0053] Each process to be spawned by the TBFE has one entry in the TBFE configuration file. In the configuration file listed above, there are entries for TPI 212, TPO 214 and the Web browser 210. Each entry specifies the location of the program file in the CMW machine 200's file system, the sensitivity label to start the program and the parameters (argument vectors, or ARGs) which should be passed to the program. The parameters define the communication channels between the different processes, such as the TCP port number that the TPI 212 should listen to and the X-server that the Web browser 210 should direct the display message to.

[0054] The TBFE configuration file includes an entry for TPI 212. The entry specifies the location of the TPI 212 program, assigns to TPI 212 the label "SYSTEM INSIDE", and declares the following parameters: "-l" defines the sensitivity label "SYSTEM MIDDLE" for TPI 212 to interact with; "s" defines the display server 232 "zhong-q-1" to be used; and "-n" defines the proxy number "1" used for communications with the display server 232. In practice, display servers are allocated port numbers running from 6000. Thus, a proxy number of "1" maps to a port number of 6001.

[0055] The configuration file also includes an entry for TPO 214. The entry specifies the location of the TPO 214 program, assigns to TPO 214 the label "SYSTEM OUTSIDE", and declares the following parameters: "-d" defines the debug level as "3"; and "-s" sends all the debug information to be displayed on "stderr"

[0056] Finally, the configuration file includes an entry for the Web browser 210. The entry specifies the location of the Web browser 210 program, assigns to the Web browser 210 the label "SYSTEM MIDDLE" and declares the parameter: "-display localhost: 1", which configures Netscape to send display messages to TPI 212, on proxy server number 1, instead of to the default X-server.

[0057] After reading the configuration file successfully, in step 515 the TBFE processes the entries one by one. For each entry in the configuration file, in step 520, the TBFE raises the Chsubjsl privilege, which allows it to adopt the sensitivity label required for the respective child process in step 525. In step 530, the TBFE drops the Chsubjsl privilege, to prevent a spawned process from misusing it. Then, in step 535, TBFE spawns the respective child process. In effect, TBFE changes its own sensitivity label to the required sensitivity label of the child process that it is going to spawn in order that the child process inherits the correct sensitivity label, as specified in the configuration file. Next, the TBFE again raises the Chsubjsl privilege in step 540, reverts to its original sensitivity label in step 545 and, finally drops the Chsubjsl privilege in step 550. This process repeats, in step 555 for all three entries in the configuration file until both proxies and the Web browser 210 have been spawned.

[0058] Finally, in step 560, the TBFE waits for one of the child processes that it spawned to terminate (this will usually be the Web browser 210 when the user has finished using it), and then, in step 565, sends exit signals to the other child processes and itself exits in step 570. Thus, TBFE acts as a single point-of-entry to the Web browser 210. Also, the TBFE will terminate the whole group of processes when any single member terminates for any reason.

[0059] Other than the Chsubjsl privilege, TBFE also needs Configaudit, Suspendaudit and Writeaudit privileges to enable it to configure, manipulate and write audit records. Audit records may be used as a historical log of events, which can be analysed to trace any unusual activity, potentially resulting from rogue mobile code. Auditing is well known in computer system management practice, and will not thus be described herein in any further detail.

[0060] When TPI 212 is started by the TBFE in the SI compartment, TPI 212 makes a system call which allows it to act as a multilevel server. To make the system call, TPI 212 requires the Chsubjsl privilege; TPI 212 raises the Chsubjsl privilege, makes the system call and then lowers the Chsubjsl privilege again. Once acting as a multilevel server, TPI 212 can receive connections on its allocated TCP port from the SM compartment as well as from the SI compartment.

[0061] When the TPO 214 is started by the TBFE in the SO compartment, TPO 214 also makes a system call which allows it to act as a multilevel server. To make the system call, as for TPI 212, TPO 214 requires the Chsubjsl privilege; TPO 214 raises the Chsubjsl privilege, makes the system call and then lowers the Chsubjsl privilege again. TPO 214 then waits for connections on its allocated TCP port which is typically port 1080, the default SOCKS port. The same port number is also defined in the Web browser 210's options to be used by the Web browser 210 as the messaging proxy port number to which all Web requests are sent.

[0062] Having started a Web browser 210 on the CMW machine 200, as described above, the user is presented with a standard Web browser 210 screen, which is rendered in an X-window of the X display server 232 on the user machine 230. The display server 232 facilitates all keyboard or mouse interaction by the user with the window by sending events to the Web browser 210 on the CMW machine 200. The Web browser 210 responds with requests, which control the display server 232, for example to update the X-window display. For ease of understanding only, both X events and X requests will be referred to as X-messages. Typically, the initial display is that of the user's 'home page'.

[0063] The sequence of steps that occur when a user requests a Web page from a Web server 252 will now be described with reference to the flow diagram in Figure 6.

[0064] In step 600, the user submits a request for a specific Web page, or other resource. The request can be a result of the user selecting a hyperlink or typing in the respective URL (universal resource locator). The request is received by TPI 212 in step 605. In step 610, TPI 212 raises the Allowmacwrite privilege, to allow TPI 212 to override the MAC's read/write restrictions, in order to transfer the request from the display server 232 (attached to the SI compartment) to the Web browser 210 (in the SM compartment) in step 615. When the transfer is complete, in step 620, TPI 212 lowers the Allowmacwrite privilege again.

[0065] In step 625, the Web browser 210 receives the request and attempts to initiate a connection with the appropriate remote Web server 252, which holds the required Web page. TPO 214 receives the connection request from the Web browser 210 in step 630 and raises the Allowmacread privilege, in step 635, in order to facilitate data transfer from the Web browser 210 (in the SM compartment) to the external network (attached to the SO compartment). Then, in step 640, TPO 214 forwards the connection request to the external network. TPO 214 also acts to filter the request to block communications with prohibited external sites. After transmission is complete, in step 645, TPO 214 lowers the Allowmacread privilege again.

[0066] The processes that occur once the request reaches the Internet are well known in the present art and will not therefore be described herein in detail. In brief, however, in step 650, the Web server 252 receives the request and responds by returning the Web page and associated mobile code to the CMW machine 200. In practice, one Web page typically references, and is rendered from, multiple data sources (commonly containing data such as formatted text and graphics images), which are down-loaded onto a Web browser using multiple HTTP requests. In the present case, where the Web page includes mobile code, there will be a reference to at least one embedded process, for example a Java applet, which is down-loaded to the Web browser in the form of byte codes.

[0067] In step 655, TPO 214 receives the stream of HTTP from the Web server 252. TPO 214 again filters the stream at this stage to block undesirable messages. Then, in step 660, TPO 214 raises the Allowmacwrite privilege and passes the HTTP stream from the SO compartment to the Web browser 210 in the SM compartment in step 664. Then, in step 667, TPO 214 lowers the Allowmacwrite privilege.

[0068] The Web browser 210 receives the stream and interprets the content as a Web page with embedded mobile code, in step 670. The Web browser 210, which is configured to allow mobile code to execute, then loads the mobile code into memory and executes it in step 674. As a result of executing the mobile code, the Web browser 210 generates a graphical output, in step 677, and requests a connection, in step 680, to pass the output X-messages to the display server 232.

[0069] When the Web browser 210 requests a connection from the SM compartment, TPI 212 accepts the request, evaluates it and tries to make a connection to the remote display server 232 in step 685. The identity and location of the remote display server 232 that the user is using is passed to TPI 212 as a parameter in the configuration file, as described above. On successfully connecting to the display server 232, TPI 212 raises the Allowmacread and Allowmacwrite privileges in step 690 and, having established a connection, pumps the messages between the SM and the SI compartments in step 695 to the display server 232. Optionally, some X-message filtering can also be performed here to prevent suspicious X-messages, potentially generated by the mobile code, from getting through.

[0070] Then, TPI 212 lowers the Allowmacread and Allowmacwrite privileges in step 697 and, finally, in step 699, the display server 232 receives the X-messages and renders the X-window appropriately.

[0071] The users of the internal network 220, who can

only connect to the SI compartment, cannot bypass the security administration by directly starting their own Web browser. This is because Web browsers started by internal users cannot gain access to the TPO 214, as a result of TPO 214 accepting connections only from the SM and SO compartments. Web browsers 210 started by internal users directly can therefore interact only with the internal network 220.

[0072] It is emphasised that the embodiment described above defines only one specific way of working the present invention, which conveniently takes advantage of HP's CMW operating system. Clearly, other CMW-compliant operating systems, such as SUN Microsystems' Trust Solaris operating system, could be readily configured to implement the invention. Indeed, embodiments of the invention could be implemented in any operating system, by configuring the operating system to provide appropriate functionality. The present invention should therefore be read broadly to encompass any system that applies the general teachings that are herein disclosed.

[0073] It will be appreciated that the invention is particularly suited to increasing security in scenarios where transactional clients are down-loaded as mobile code for interaction with other clients or servers in a client-server environment. Such an environment can be one that complies with the CORBA (Common Object Request Broker Architecture) model.

References:

[0074]

[NCSC]: National Computer Security Centre, "Department of Defence Trusted Computer System Evaluation Criteria", DoD Standard 5200 28-STD, 1985

[DIA 91]: "Compartmented Mode Workstation Evaluation Criteria VERSION 1 (Final)", J.P.L. Woodward, DDS-2600-6243-91, 1991.

[X Window]: "X Window System", Scheifler, Robert and James Gettys, Digital Press, 1992

[SOCKS]: "SOCKS Protocol Version 5", M. Leech, M. Ganis, Y Lee, etc., RFC 1928, March 1996

CMW machine 200 Manuals:

[0075]

HP-UX Trusted OS Installation Manual
HP-UX Trusted OS Read Me First/ Release Notes
HP-UX 10.09.01 CMW machine 200 Trusted Facilities Manual
HP-UX 10.09.01 CMW machine 200 Key Security Concepts
HP-UX 10.09.01 CMW machine 200 Support Media User's Guide
HP-UX 10.09.01 CMW machine 200 Trusted Facility Admin Ref. Manual
HP-UX 10.09.01 CMW machine 200 MaxSix Administrator's Guide
HP-UX 10.09.01 CMW machine 200 Security Features User's Guide
HP-UX 10.09.01 CMW machine 200 Security Features Programmer's Guide

Claims

1. A browser system, comprising:

   a browser process configured to receive from a remote data source a resource incorporating mobile code and to process the mobile code to generate graphical output data; and
   an inside interface process configured to provide a communications channel between the browser process and a remote display system to facilitate transfer of the graphical output data to the remote display system.

2. A browser system according to claim 1, comprising an operating system which associates processes or objects within the operating environment of the operating system with one of a number of sensitivity labels, wherein the browser process has a first sensitivity label and data associated with the remote display system has a second sensitivity label.

3. A browser system according to either preceding claim, wherein the inside interface process has a first privilege which allows it to transfer data from the browser process to the remote display system.

4. A browser system according to claim 3, wherein the inside interface process is configured to raise the first privilege when data transfer is required and lower the first privilege after data transfer is completed.

5. A browser system according to any one of the preceding claims, further comprising an outside interface process, which provides a communications channel between the browser process and the remote data source to facilitate transfer of data from the remote data source to the browser process.

6. A browser system according to claim 5, wherein data associated with the remote data source has a third sensitivity label

7. A browser system according to claim 5 or claim 6, wherein the outside interface process has a second privilege which allows it to transfer data from the remote data source to the browser process.

8. A browser system according to claim 7, wherein the outside interface process is configured to raise the second privilege when data transfer is required and lower the second privilege after data transfer is completed.

9. A browser system according to claim 1, wherein the inside interface process is configured as a multi-level process whereby it can receive connection requests having either the first sensitivity label or the second sensitivity label.

10. A browser system according to claim 5, wherein the outside interface process is configured as a multi-level process whereby it can receive connection requests having either the second sensitivity label or the third sensitivity label.

11. A browser system according to any one of the preceding claims, wherein the operating system enforces Mandatory Access Control.

12. A browser system configured for operation in an operating system enforcing mandatory access control, the browser system comprising:

a browser process having a first sensitivity label,
an inside interface process having privileges that allow it to transfer data between the browser process and a display system, the operating system being configured to allocate data associated with the remote display system with a second sensitivity label; and
an outside interface process having privileges that allow it to transfer data between the browser and a remote data source, the operating system being configured to allocate data associated with the remote data source with a third sensitivity label, the browser process being configured to:
receive via the outside interface process a resource including mobile code;
process the mobile code to provide graphical output data; and
send the graphical output data via the inside interface process to the display server.

13. A browser system according to claim 11, wherein the browser process is further configured to receive a request, via the inside interface process, from the display server for a remote resource including mobile code and to transfer a respective request for the resource, via the outside interface process, to the remote data source.

14. A method of securely accessing a resource including mobile code using a browser system configured for operation in an operating system enforcing mandatory access control, the method including a browser process having a first sensitivity label enacting the steps of:

receiving, via an outside interface process, a resource including mobile code from a remote data source, the operating system being configured to allocate data associated with the remote data source with a third sensitivity label and the outside interface process having privileges that allow it to transfer data between the browser and the remote data source;
processing the mobile code to provide graphical output data; and
sending the graphical output data via an inside interface process to a display server, the operating system being configured to allocate data associated with the remote display system with a second sensitivity label and the inside interface process having privileges that allow it to transfer data between the browser process and the display system.

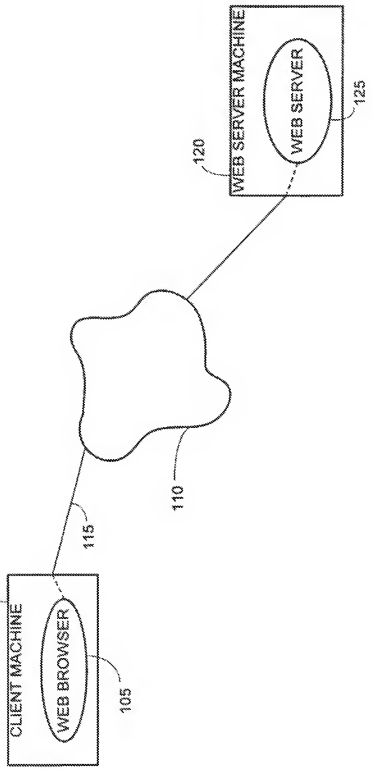

100
CLIENT MACHINE
WEB BROWSER
105
115
110
120
WEB SERVER MACHINE
WEB SERVER
125

FIGURE 1.

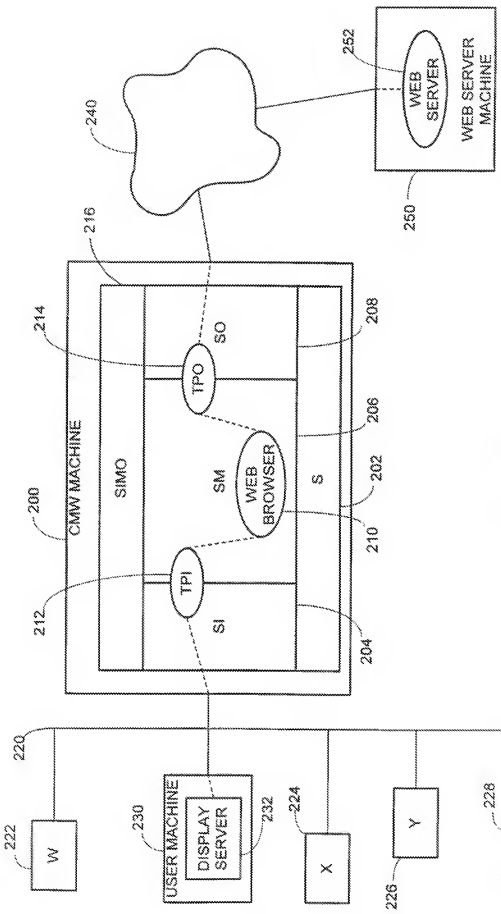

CMW MACHINE
200
SIMO
SM
S
SI
TPO
TPI
WEB BROWSER
SO
202
204
206
208
210
212
214
216
240
WEB SERVER
WEB SERVER MACHINE
250
252
220
W
222
USER MACHINE
DISPLAY SERVER
230
232
X
224
Y
226
Z
228

FIGURE 2.

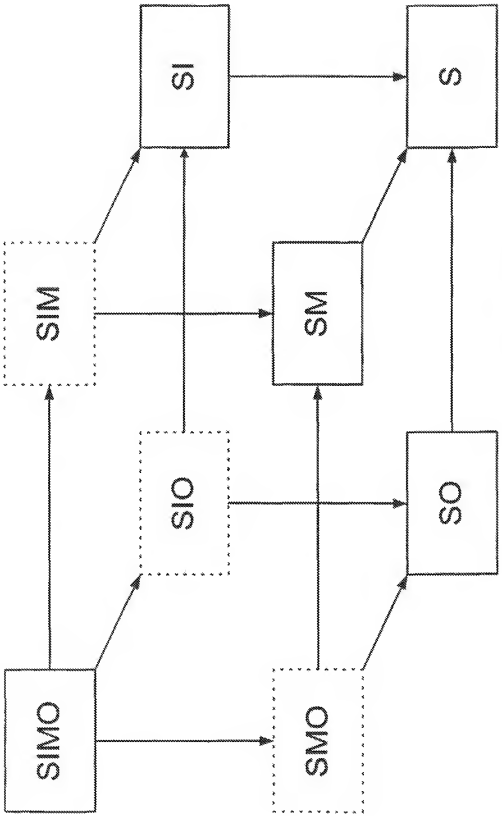


FIGURE 3.

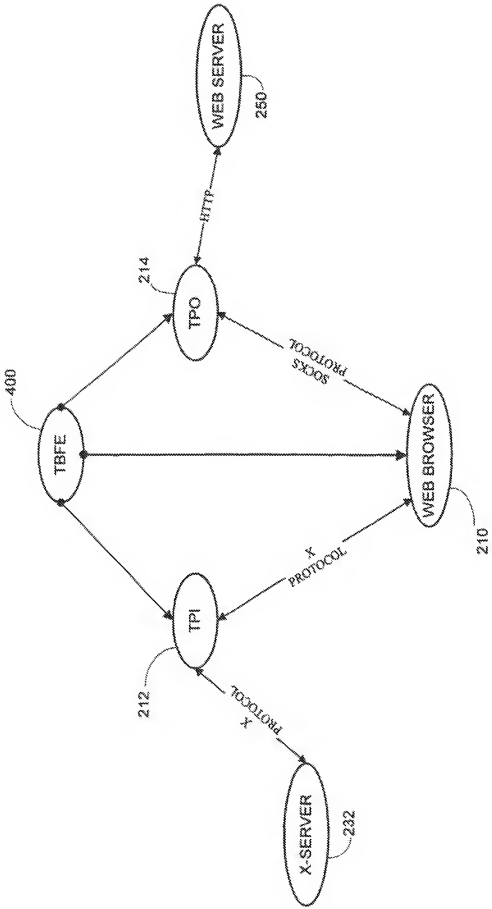

WEB SERVER
250
HTTP
214
TPO
SOCKS PROTOCOL
400
TBFE
WEB BROWSER
210
X PROTOCOL
TPI
212
PROTOCOL X
X-SERVER
232

FIGURE 4.

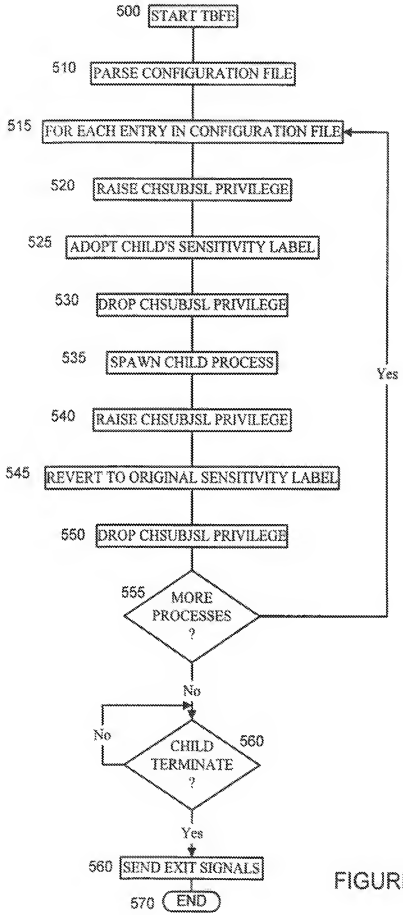

500
START TBFE
510
PARSE CONFIGURATION FILE
515
FOR EACH ENTRY IN CONFIGURATION FILE
520
RAISE CHSUBJSL PRIVILEGE
525
ADOPT CHILD'S SENSITIVITY LABEL
530
DROP CHSUBJSL PRIVILEGE
535
SPAWN CHILD PROCESS
540
RAISE CHSUBJSL PRIVILEGE
545
REVERT TO ORIGINAL SENSITIVITY LABEL
550
DROP CHSUBJSL PRIVILEGE
555
MORE PROCESSES ?
Yes
No
No
CHILD TERMINATE ?
560
Yes
560
SEND EXIT SIGNALS
570
END

FIGURE 5.

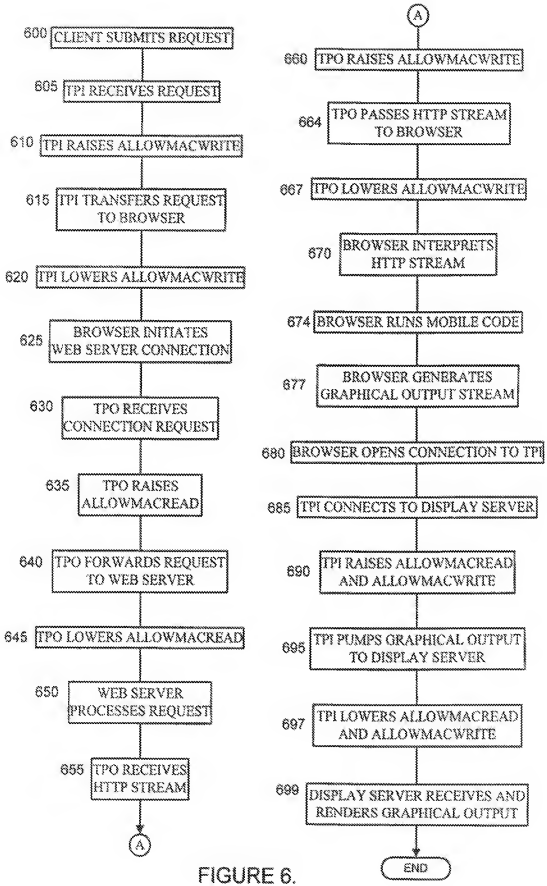

600 CLIENT SUBMITS REQUEST
605 TPI RECEIVES REQUEST
610 TPI RAISES ALLOWMACWRITE
615 TPI TRANSFERS REQUEST TO BROWSER
620 TPI LOWERS ALLOWMACWRITE
625 BROWSER INITIATES WEB SERVER CONNECTION
630 TPO RECEIVES CONNECTION REQUEST
635 TPO RAISES ALLOWMACREAD
640 TPO FORWARDS REQUEST TO WEB SERVER
645 TPO LOWERS ALLOWMACREAD
650 WEB SERVER PROCESSES REQUEST
655 TPO RECEIVES HTTP STREAM
A
A
660 TPO RAISES ALLOWMACWRITE
664 TPO PASSES HTTP STREAM TO BROWSER
667 TPO LOWERS ALLOWMACWRITE
670 BROWSER INTERPRETS HTTP STREAM
674 BROWSER RUNS MOBILE CODE
677 BROWSER GENERATES GRAPHICAL OUTPUT STREAM
680 BROWSER OPENS CONNECTION TO TPI
685 TPI CONNECTS TO DISPLAY SERVER
690 TPI RAISES ALLOWMACREAD AND ALLOWMACWRITE
695 TPI PUMPS GRAPHICAL OUTPUT TO DISPLAY SERVER
697 TPI LOWERS ALLOWMACREAD AND ALLOWMACWRITE
699 DISPLAY SERVER RECEIVES AND RENDERS GRAPHICAL OUTPUT
END

FIGURE 6.

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 97 30 9328

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	WALLACH D S ET AL: "Extensible security architectures for Java" 16TH ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, SAINT MALO, FRANCE, 5-8 OCT. 1997, vol. 31, no. 5, ISSN 0163-5980, OPERATING SYSTEMS REVIEW, DEC. 1997, ACM, USA, pages 116-128, XP002059825 * the whole document *	1,12,14	G06F17/30
A	ROUAIX F: "A Web navigator with applets in Caml" COMPUTER NETWORKS AND ISDN SYSTEMS, vol. 11, no. 28, May 1996, page 1365-1371 XP004018234 * the whole document *	1,12,14	
A	GOSLING J ET AL: "THE JAVA LANGUAGE ENVIRONMENT. A WHITE PAPER" SUN DELIVERS JAVA WORKSHOP, October 1995, pages 1, 4-85, XP002042922 * page 14, line 1 - page 14, line 15 *	1,12,14	TECHNICAL FIELDS SEARCHED (Int.Cl.6) G06F

This present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	23 March 1998	Katerbau, R

CATEGORY OF CITED DOCUMENTS

X particularly relevant if taken alone
Y particularly relevant if combined with another document of the same category
A technological background
O non-written disclosure
P intermediate document

T theory or principle underlying the invention
E earlier patent document, but published on, or after the filing date
D document cited in the application
L document cited for other reasons

& member of the same patent family, corresponding document